Promoting Data Science

Volume 2 2022 Number 2

Journal of Behavioral Data Science V2N2 (2022)

https://isdsa.org

JOURNAL OF BEHAVIORAL DATA SCIENCE

Editor

Zhiyong Zhang, University of Notre Dame, USA

Associate Editors

Denny Borsboom, University of Amsterdam, Netherlands

Hawjeng Chiou, National Taiwan Normal University, Taiwan

Ick Hoon Jin, Yonsei University, Korea

Hongyun Liu, Beijing Normal University, China

Christof Schuster, Giessen University, Germany

Jiashan Tang, Nanjing University of Posts and Telecommunications, China

Satoshi Usami, University of Tokyo, Japan

Ke-Hai Yuan, University of Notre Dame, USA

ISDSA

ISBN: 2575-8306 (Print) 2574-1284 (Online)

https://jbds.isdsa.org

JOURNAL OF BEHAVIORAL DATA SCIENCE

Guest Editors

Tessa Blanken, University of Amsterdam, Netherlands

Alexander Christensen, University of Pennsylvania, USA

Han Du, University of California, Los Angeles, USA

Hudson Golino, University of Virginia, USA

Timothy Hayes, Florida International University, USA

Suzanne Jak, University of Amsterdam, Netherlands

Ge Jiang, University of Illinois at Urbana-Champaign, USA

Zijun Ke, Sun Yat-Sen University, China

Mark Lai, University of Southern California

Haiyan Liu, University of California, Merced, USA

Laura Lu, University of Georgia, USA

Yujiao Mai, ISDSA, USA

Ocheredko Oleksandr, Vinnytsya National Pirogov Memorial Medical University, Ukraine

Robert Perera, Virginia Commonwealth University, USA

Sarfaraz Serang, Utah State University, USA

Xin (Cynthia) Tong, University of Virginia, USA

Riet van Bork, University of Pittsburgh, USA

Qian Zhang, Florida State University, USA

Editorial Assistants

Wen Qu, University of Notre Dame, USA

Anqi Fa and Fei Gao, Nanjing University of Posts and Telecommunications, China

No Publication Charge and Open Access

jbds@isdsa.org

List of Articles

Journal of Behavioral Data Science, 2022, *2* (2), 1–22.
DOI: `https://doi.org/10.35566/jbds/v2n2/p1`

Disentangling the Influence of Data Contamination in Growth Curve Modeling: A Median Based Bayesian Approach

Tonghao Zhang[1], Xin Tong*[2], and Jianhui Zhou[1]

[1] Department of Statistics, University of Virginia
[2] Department of Psychology, University of Virginia
`xt8b@virginia.edu`

Abstract. Growth curve models (GCMs), with their ability to directly investigate within-subject change over time and between-subject differences in change for longitudinal data, are widely used in social and behavioral sciences. While GCMs are typically studied with the normal distribution assumption, empirical data often violate the normality assumption in applications. Failure to account for the deviation from normality in data distribution may lead to unreliable model estimation and misleading statistical inferences. A robust GCM based on conditional medians was recently proposed and outperformed traditional growth curve modeling when outliers were present resulting in nonnormality. However, this robust approach was shown to perform less satisfactorily when leverage observations existed. In this work, we propose a robust double medians growth curve modeling approach (DOME GCM) to thoroughly disentangle the influence of data contamination on model estimation and inferences, where two conditional medians are employed for the distributions of the within-subject measurement errors and of random effects, respectively. Model estimation and inferences are conducted in the Bayesian framework, and Laplace distributions are used to convert the optimization problem of median estimation into a problem of obtaining the maximum likelihood estimator for a transformed model. A Monte Carlo simulation study has been conducted to evaluate the numerical performance of the proposed approach, and showed that the proposed approach yields more accurate and efficient parameter estimates when data contain outliers or leverage observations. The application of the developed robust approach is illustrated using a real dataset from the Virginia Cognitive Aging Project to study the change of memory ability.

Keywords: Robust methods · Growth curve modeling · Conditional medians · Laplace distribution

1 Introduction

Longitudinal data track the same subjects across different time points. In contrast to cross-sectional data, longitudinal data allow for measuring the within-subject change over time, capturing the duration of events, and recording the timing of various events. Growth curve modeling is one of the most frequently used analytical techniques for longitudinal data analysis (e.g., McArdle & Nesselroade, 2014), due to its abilities to examine within-subject change over time, and to investigate into differences of the change patterns among individuals.

In growth curve modeling, estimation methods that are based on the normality assumption in data distribution are widely accepted, and have been incorporated in many statistical software packages. When data all come from a normal population, those methods are able to provide consistent and efficient parameter estimators. However, practical data are often contaminated with outlying observations in social and behavioral sciences, so that the normality assumption is violated in real data analysis. For example, Micceri (1989) investigated 440 large-scale data sets in psychology and found that almost all of them were significantly nonnormal. When the normality assumption does not hold, traditional growth curve modeling which focuses on conditional means of the outcome variables may lead to inefficient and even biased model estimation (e.g., Yuan & Bentler, 2001).

To disentangle the influence of data contamination, the cause of the contamination needs to be understood. Reflected in growth curve modeling, data contamination may be caused by extreme scores in either random effects or within-subject measurement errors. The former is referred to as leverage observations and the latter is called outliers (Tong & Zhang, 2017). It is necessary to distinguish these two types of outlying observations since their influences on the estimation of growth curve models (GCMs) are different. Although techniques to detect leverage observations and outliers have been developed (e.g., Tong & Zhang, 2017), it has been shown that outlying observation detection in longitudinal data is a challenging problem whose sensitivity and specificity are difficult to guarantee. Even if the leverage observations and outliers are correctly identified, simply deleting them could result in decreased statistical efficiency (Lange, Little, & Taylor, 1989).

To address the issue of data contamination, various robust estimation methods have been proposed to produce reliable analysis in the presence of data nonnormality. Some of them rely on making distributional assumptions that are more reasonable to the dataset, such as using Student's t distributions or mixture of normal distributions (Lu & Zhang, 2014; Reich, Bondell, & J., 2010; Tong & Zhang, 2012). However, those methods are sensitive to the choice of the assumed distribution, which is difficult to specifiy *a priori* and verify afterwards, especially for small sized data. Another genre of robust methods assign weights to observations according to their distances from the center of the majority of data so that extreme cases are downweighted (e.g., Pendergast & Broffitt, 1985; Singer & Sen, 1986; Yuan & Bentler, 1998a; Zhong & Yuan, 2010). Those weighting methods may have limitations under certain conditions, e.g., in general, they do not take the leverage observations into consideration (Zhong & Yuan, 2011).

Median-based regression and its generalization, quantile regression (Koenker, 2004), have emerged as another genre of robust methods. Such methods are distribution free, and have been extended to many topics such as penalized regression and time series models. Although the median-based methods are still not widely applied to longitudinal research (Geraci, 2014), they are getting more and more attention (e.g., Cho, Hong, & Kim, 2016; Galvao & Poirier, 2019; Huang, 2016; Smith, Fuentes, Gordon-Larsen, & Reich, 2015; Zhang, Huang, Wang, Chen, & Langland-Orban, 2019). Recently a robust growth curve modeling approach using conditional medians was proposed (Tong, Zhang, & Zhou, 2021). Although this robust approach outperformed traditional conditional mean-based growth curve modeling in the presence of outliers, it still yielded biased parameter estimates when leverage observations exist.

It is crucial to have a robust estimator for growth curve models when data are contaminated with both outliers and leverage observations in longitudinal studies. To obtain such an estimator, we develop a **DO**uble **ME**dian-based structure (DOME) to mitigate potential distortion in both distributions of random effects and of within-subject measurement errors in growth curve modeling. When random effects and measurement errors are symmetrically distributed, the estimates based on the developed method will be very close to the ones obtained by traditional growth curve modeling estimation method. It is expected that DOME growth curve modeling is more robust against nonnormal data than traditional conditional mean-based method, and also outperforms the median-based growth curve modeling in Tong et al. (2021). Bayesian methods are used for DOME GCM estimation because they can conveniently infer parameters that do not have symmetric distributions (e.g., variance parameters), incorporate prior information to make parameter estimates more efficient, naturally accommodate missing data without requiring new techniques, and are powerful to deal with complex model structures.

In sum, the purpose of this work is to develop a robust Bayesian growth curve modeling approach that is effective to analyze longitudinal data that are contaminated with both outliers and leverage observations in general. In the following sections, the idea of the proposed robust approach, DOME GCM, will be introduced, Monte Carlo simulation studies are conducted to evaluate the numerical performance of the developed method and compare its performance with those of traditional growth curve modeling and the median-based method developed by Tong et al. (2021), and an empirical example is provided to illustrate the application of DOME GCM to study the change of memory scores using a real dataset from the Virginia Cognitive Aging Project (Salthouse, 2014, 2018). We conclude this article with discussions and suggestions on future research directions.

2 DOME Growth Curve Modeling

In longitudinal studies, the same subjects are measured repeatedly over time. Suppose that a longitudinal study is conducted on a cohort of individuals, indexed by $i = 1, ..., N$. Let $\boldsymbol{y}_i = (y_{i1}, ..., y_{iT_i})^\intercal$ be a $T_i \times 1$ vector, where y_{it} is the observation on individual i at time t for $t = 1, ..., T_i$ with T_i being the maximum follow-up time for this individual. A typical form of the unconditional GCMs can be formulated as

$$\begin{aligned} \boldsymbol{y}_i &= \boldsymbol{X}_i \boldsymbol{b}_i + \boldsymbol{\epsilon}_i, \\ \boldsymbol{b}_i &= \boldsymbol{\beta} + \boldsymbol{u}_i, \end{aligned} \tag{1}$$

where $\boldsymbol{X}_i$ is a $T_i \times q$ factor loading matrix recording the time of measurements. It can be different across individuals when they are not measured at a common set of time. The vector $\boldsymbol{b}_i$ is a $q \times 1$ vector of random effects, and $\boldsymbol{\epsilon}_i$ is a vector of within-subject measurement errors. The vector of random effects $\boldsymbol{b}_i$ varies across individuals, and $\boldsymbol{\beta}$ represents the fixed effects for the population. The residual vector $\boldsymbol{u}_i$ represents the random component of $\boldsymbol{b}_i$. Without loss of generality, we assume the number of measurement occasions to be the same for all individuals, i.e., $T_i \equiv T$.

Traditional GCMs typically assume that both $\boldsymbol{\epsilon}_i$ and $\boldsymbol{u}_i$ follow multivariate normal (MN) distributions,

$$\begin{aligned} \boldsymbol{\epsilon}_i &\sim \mathrm{MN}_T(\boldsymbol{0}, \boldsymbol{\Phi}), \\ \boldsymbol{u}_i &\sim \mathrm{MN}_q(\boldsymbol{0}, \boldsymbol{\Sigma}), \end{aligned}$$

where the subscripts of MN distributions imply the dimensionalities of the random vectors. The covariance matrix $\boldsymbol{\Phi}$ is usually assumed to be diagonal $\boldsymbol{\Phi} = \sigma_\epsilon^2 \boldsymbol{I}$, indicating that measurement errors have equal variance and are independent across different time points.

Traditional GCMs focus on modeling the conditional means of the outcome variables, $E(\boldsymbol{y}_i|\boldsymbol{b}_i) = \boldsymbol{X}_i \boldsymbol{b}_i$, and estimating the common growth parameters, $E(\boldsymbol{b}_i) = \boldsymbol{\beta}$.

However, it is well known that mean is sensitive to outlying observations. Tong et al. (2021) proposed a median-based GCM where the conditional medians of the outcome variables $Q_{0.5}(\boldsymbol{y}_i|\boldsymbol{b}_i)$, are examined instead of the conditional means $E(\boldsymbol{y}_i|\boldsymbol{b}_i)$. Their numerical results showed that this approach is only robust against outliers, but not against leverage observations. This is as expected because an outlier is caused by an extreme score in $\boldsymbol{\epsilon}_i$ and a leverage observation is caused by an extreme score in $\boldsymbol{u}_i$. The robust approach in Tong et al. (2021) only models the conditional medians of the outcome variables at the level-one model. Although it seems to be a natural extension to model conditional medians of the level-two model as well to address the influence of leverage observations, the extension is not straightforward because within-subject measurement errors $\boldsymbol{\epsilon}_i$ were assumed to be independent across different time points and can be modeled as univariate random variables, whereas $\boldsymbol{u}_i$ has to be specified as a multivariate variable with dependent components.

In this paper, we tackle the complicated multivariate problem and propose a robust method by adopting two median structures as alternatives, $Q_{0.5}(\boldsymbol{y}_i|\boldsymbol{b}_i)$ replacing $E(\boldsymbol{y}_i|\boldsymbol{b}_i)$ and $Q_{0.5}(\boldsymbol{b}_i)$ replacing $E(\boldsymbol{b}_i)$. We call this new model a double medians growth curve model (DOME GCM).

2.1 DOME GCM specification

As discussed previously, outlying observations in longitudinal data can be either *outliers* as a result of extreme measurement errors, or *leverage observations* due to extreme scores in random effects (Tong & Zhang, 2017). The proposed DOME GCM aims to handle the presence of data nonnormality due to both types of outlying observations.

DOME growth curve modeling is an extension of traditional mean-based method,

$$\begin{aligned}
&\boldsymbol{y}_i = \boldsymbol{X}_i\boldsymbol{b}_i + \boldsymbol{\epsilon}_i,\\
&\boldsymbol{b}_i = \boldsymbol{\beta} + \boldsymbol{u}_i,\\
&Q_{0.5}(\boldsymbol{\epsilon}_i|\boldsymbol{u}_i) = \boldsymbol{0},\\
&Q_{0.5}(\boldsymbol{u}_i) = \boldsymbol{0},
\end{aligned} \tag{2}$$

where medians for vector are taken entry-wise. In this multilevel modeling framework, at the first level, the relationship between $\boldsymbol{X}_i$ and the outcome variable $\boldsymbol{y}_i$ is based on the conditional median function $Q_{0.5}(\boldsymbol{\epsilon}_i|\boldsymbol{u}_i) = \boldsymbol{0}$. At the second level, random effect $\boldsymbol{b}_i$ varies around the median $\boldsymbol{\beta}$, the fixed effects for the population. The random residuals $\boldsymbol{u}_i = [u_{i1}, u_{i2}, \ldots, u_{iq}]^\intercal$ are the random components of $\boldsymbol{b}_i$. Since no distributional assumption is imposed on $\boldsymbol{\epsilon}_i$ or $\boldsymbol{u}_i$, the proposed DOME growth curve model is distribution-free.

The multilevel structure in Equation (2) can be expressed compactly as

$$y_{it} = \boldsymbol{x}_{it}^\intercal\boldsymbol{\beta} + \boldsymbol{x}_{it}^\intercal\boldsymbol{u}_i + \epsilon_{it}, \tag{3}$$

where $Q_{0.5}(\epsilon_{it}|\boldsymbol{u}_i) = 0$ and $Q_{0.5}(\boldsymbol{u}_i) = 0$. Here $\boldsymbol{x}_{it}$ is the transpose of the tth row of $\boldsymbol{X}_i$. The population regression coefficient $\boldsymbol{\beta}$ is the main parameter of interest in the DOME GCM model.

2.2 Estimation of the DOME GCM

Recall that in traditional regression based on conditional means, we minimize the sum of squared residuals to estimate model parameters. Similarly, in a median-based regression,

$$y_i = \boldsymbol{x}_i^\intercal\boldsymbol{\beta} + \epsilon_i, Q_{0.5}(\epsilon) = 0,$$

estimation is carried out by minimizing the sum of absolute residuals,

$$\hat{\boldsymbol{\beta}}_{0.5} = \underset{\boldsymbol{\beta}}{argmin} \sum_{i=1}^{N} |y_i - \boldsymbol{x}_i^\intercal\boldsymbol{\beta}|. \tag{4}$$

However, this minimization involves the sum of absolute values, which is not differentiable at zero, meaning that explicit solutions to the minimization problem are unavailable. Moreover, when more than one median constraints are introduced, as in the case of the DOME GCM model in Equations (2), defining an objective function similar to Equation (4) becomes difficult, for the reason that the objective function should be marginalized and involves integration. The computational challenge can be overcome by introducing Laplace distributions to make a connection between the estimation of DOME GCM in Equations (2) and the maximum likelihood principle (Geraci, 2014).

The Laplace distribution has a relationship with the l_1-norm loss function described in Koenker and Bassett (1978). This relationship is best demonstrated by the probability density function for a unidimensional Laplace distribution $X \sim Laplace(\mu, \sigma)$,

$$p(x|\mu, \sigma) = \frac{1}{2\sigma} exp\left\{-\frac{1}{\sigma}|x-\mu|\right\},$$

where $\mu \in R$ is the location parameter and $\sigma \in R_+$ is the scale parameter. The mean and variance of the distribution are given by

$$E(X) = \mu,$$
$$Var(X) = 2\sigma^2,$$

respectively. Laplace distribution is also known as the standard double exponential distribution.

The univariate Laplace distribution can be extended to the multivariate Laplace distribution (Kozubowski & Podgorski, 2000). The marginal distributions of a multivariate Laplace distribution variable are unidimensional Laplace distributions. A multivariate Laplace distribution is parameterized by location $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, denoted as $\boldsymbol{Y} \sim Laplace(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. For a n-dimensional Laplace distribution, if $\boldsymbol{\mu} = 0$, the probability density function of the multivariate Laplace distribution is given by

$$p(\boldsymbol{y}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{2}{(2\pi)^{n/2}|\boldsymbol{\Sigma}|^{0.5}} \left(\frac{\boldsymbol{y}^\intercal \boldsymbol{\Sigma}^{-1} \boldsymbol{y}}{2}\right)^{v/2} K_v(2\sqrt{\boldsymbol{y}^\intercal \boldsymbol{\Sigma}^{-1} \boldsymbol{y}}),$$

where $v = \frac{2-n}{2}$ and K_v is the modified Bessel function of the second kind.

We employ Laplace distributions to convert the problem of estimating DOME GCM into a problem of obtaining the maximum likelihood estimator (MLE) for a transformed model. For the purpose of demonstration, we focus on a linear GCM in this paper, so that the random effect is two-dimensional

$$\boldsymbol{b}_i = \begin{bmatrix} L_i \\ S_i \end{bmatrix},$$

where L_i is the initial level and S_i is the rate of change over time for the ith individual, respectively. The transformed model for the DOME GCM in Equations

(2) is

$$\begin{aligned} y_{it} &= \boldsymbol{x}_{it}^{\intercal}\boldsymbol{b}_i + \epsilon_{it}, \\ \boldsymbol{b}_i &= \boldsymbol{\beta} + \boldsymbol{u}_i, \\ \epsilon_{it} &\sim Laplace(0, \sigma_\epsilon), \\ \boldsymbol{u}_i &\sim Laplace(\boldsymbol{0}, \boldsymbol{\Sigma}). \end{aligned} \tag{5}$$

Note that the median structures are applied for the measurement errors using a univariate Laplace distribution, and for the random effects using a bivariate Laplace distribution. Since the median of a Laplace distribution is the location parameter μ, it can be verified that

$$Q_{0.5}(\epsilon_{it}|\boldsymbol{u}_i) = 0 \quad \text{and} \quad Q_{0.5}(\boldsymbol{u}_i) = 0,$$

so that parameter estimation of DOME GCM in Equations (2) can be obtained by estimating the transformed model in Equations (5), for which the likelihood function for T observations across N subjects is

$$\begin{aligned} L(\boldsymbol{\beta}, \sigma; \boldsymbol{y}) &= \int \cdots \int \left(\prod_{i=1}^{N} p(\boldsymbol{y}_i|\boldsymbol{b}_i, \boldsymbol{\beta}, \sigma_\epsilon) \times p(\boldsymbol{b}_i|\boldsymbol{\beta}, \Sigma) \right) d\boldsymbol{b}_1 \ldots d\boldsymbol{b}_N \\ &\propto \int \cdots \int \left(\prod_{i=1}^{N} exp\left\{ -\frac{1}{\sigma_\epsilon^2} \sum_{t=1}^{T} |y_{it} - \boldsymbol{x}_i^{\intercal}\boldsymbol{b}_i| \right\} p(\boldsymbol{b}_i|\boldsymbol{\beta}, \Sigma) \right) d\boldsymbol{b}_1 \ldots d\boldsymbol{b}_N, \end{aligned} \tag{6}$$

where $p(\boldsymbol{y}_i|\boldsymbol{b}_i, \boldsymbol{\beta}, \sigma_\epsilon)$ is the conditional probability density function of $\boldsymbol{y}_i$ and $p(\boldsymbol{b}_i|\boldsymbol{\beta}, \Sigma)$ is the probability density function for multivariate Laplace distribution.

The solution of the maximum likelihood problem is difficult to derive analytically, or numerically under the frequentist framework, as $\boldsymbol{b}_i$'s need to be integrated out. Alternatively, the estimation can be carried out naturally under the Bayesian framework, as Bayesian methods with data augmentation techniques are flexible and computationally more powerful in such settings. Monte Carlo Markov Chain (MCMC) algorithms can be applied here, using empirical integration to approximate the exact integration. The basic idea of Bayesian methods is to obtain the posterior distributions of model parameters based on the likelihood function and the priors. Since the Laplace distribution can be constructed using a normal distribution and an exponential distribution, the data augmentation technique is used here to simplify the procedure to obtain posterior distributions. Specifically, to simulate a Laplace distribution with location $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$, we can generate, independently, two augmented variables $W \sim exp(1)$ and $\boldsymbol{X} \sim N(0, \boldsymbol{\Sigma})$. As a result, the variable

$$\boldsymbol{y} = \sqrt{W}\boldsymbol{X} + W\boldsymbol{\mu}$$

follows the $Laplace(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ distribution. The augmented representation provides an efficient method to draw MCMC from the posterior distribution. Particularly, if conjugate priors are used, we can derive conditional posterior distribution for the model parameters. Gibbs sampling then can be utilized, where samples of parameters are drawn iteratively from the conditional posterior distribution. This way we obtain the empirical marginal distribution of model parameters, with which model estimation and statistical inference can be performed. Noninformative conjugate priors are used in our study because of their advantage in easy Gibbs sampling derivation. Other priors, especially informative priors when previous information is available, can also be used and may be more advantageous, on potentially reducing convergence issue or decreasing computation time (e.g., Depaoli, Liu, & Marvin, 2021).

3 Performance Evaluation of DOME GCM through a Simulation Study

In this section, a simulation study is conducted to evaluate the numerical performance of the robust Bayesian DOME growth curve modeling in analyzing contaminated data with outliers and/or leverage observations, which correspond to extreme scores in measurement errors and random effects, respectively. Comparisons are drawn among the developed DOME GCM, traditional growth curve modeling based on conditional means, as well as the robust Bayesian method in Tong et al. (2021) where the median structure is only applied in the first level of GCM, referred to as the median-based method hereafter.

To directly compare with the study in Tong et al. (2021), we follow their simulation design and focus on the linear GCM as discussed in the previous section. The number of measurement occasions is set at 5, the population parameter values for the fixed effects are set as $\boldsymbol{\beta} = (\beta_L, \beta_S)^\intercal = (6.2, 1.5)^\intercal$, the variance of latent intercept $\sigma_L^2 = 0.5$, the variance of latent slope $\sigma_S^2 = 0.1$, the covariance between intercept and slope is 0, and the measurement error variance $\sigma_\epsilon^2 = 0.1$.

In the simulation, we vary the sample size ($N = 200, 500$), the percentage of outlying observations ($10\%, 25\%$), and the types of outlying observations (outliers and leverage observations). Given a specific sample size and the percentage of outlying observations $r\%$, we first generate normally distributed measurement errors $\boldsymbol{\epsilon}_i \sim MN_5(0, \sigma_\epsilon^2 \boldsymbol{I})$ and random effects $\boldsymbol{u_i} \sim MN_2(0, \Phi)$. Then $r\%$ of subjects are randomly selected to be contaminated by outlying observations in three scenarios; all selected subjects are contaminated by outliers, all selected subjects are contaminated by leverage observations, and the selected subjects are randomly contaminated with outliers or leverage observations with equal probabilities.

To generate outliers, we randomly select 2 out of the 5 observations for one subject, and replace them by data generated from $N(0, 0.1)$. To generate leverage observations, the random slopes of the contaminated subjects are set to follow the distribution $N(-3, 0.1)$, instead of $N(1.5, 0.1)$ for the population.

For each data condition, a total of 500 datasets are generated. Each dataset is analyzed using the three methods. Traditional growth curve modeling with normality assumptions is conducted under the structural equation modeling framework, using the "lavaan" package in R (Rosseel, 2012). The median-based method and the proposed DOME growth curve modeling are implemented with the "rstan" package (Stan Development Team, 2019). The Markov chain length is set to be 15,000, and the burn-in period is 7,500. A set of commonly used priors are specified for model parameters. A multivariate normal distribution prior is assumed for $\boldsymbol{\beta}$. The measurement error variance σ_ϵ^2 is given an inverse gamma prior, and inverse Wishart distribution is assumed for the covariance of random effect $\boldsymbol{\Sigma}$. More details can be found in the R code for implementation in the appendix.

3.1 Evaluation criteria

We obtain parameter estimation based on the three methods. Estimation bias, empirical standard error (ESE), average standard error (ASE), and mean squared error (MSE) for each parameter are calculated and used to evaluate the numerical performances of those methods. Let θ denote a parameter and also its population value, and let $\hat{\theta}_k$ and SE_k denote its estimate and the corresponding estimated standard error in the kth replication. Then the parameter estimate of θ, $\hat{\theta}$, is calculated as the average of parameter estimates of 500 simulation replications

$$\hat{\theta} = \frac{1}{500}\sum_{k=1}^{500}\hat{\theta}_k.$$

The bias of $\hat{\theta}$ is $bias(\hat{\theta}) = \hat{\theta} - \theta$. The empirical standard error is defined by

$$ESE(\hat{\theta}) = \sqrt{\frac{1}{499}\sum_{k=1}^{500}(\hat{\theta}_k - \hat{\theta})^2}.$$

The average standard error is

$$ASE(\hat{\theta}) = \frac{1}{500}\sum_{k=1}^{500} SE_k.$$

When standard errors are estimated accurately in model estimation by the developed method, ASE should be very close to ESE. The mean squared error is calculated by $MSE(\hat{\theta}) = bias^2 + ESE^2$. A smaller MSE indicates a more accurate and precise estimator.

When Bayesian methods are applied, Geweke tests (Geweke, 1992) are used to assess the convergence of Markov chains for all simulation replications. After the burn-in period, if sample parameter values are drawn from the stationary distribution of the chain, the means of the first and last parts of the Markov chain (by convention the first 10% and the last 50 %) should be equal, and the Geweke statistic asymptotically follows a standard normal distribution. We report the convergence percentage of the 500 replications by Geweke test, and the summarized model estimation results are based only on converged replications.

In practice, if the MCMC procedure does not converge, we may adopt longer Markov chains, or choose different starting values or prior distributions to yield convergent Markov chains.

The model estimation time, in the number of seconds, is reported for the two Bayesian methods. The median estimation time (MET) is the median of the estimation time for the converged replications.

3.2 Results

When there are no outlying observations, traditional mean-based growth curve modeling and the two robust median based growth curve modeling approaches perform equally well.

Tables 1 - 2 summarize the parameter estimation results for the overall latent slopes (β_S) and the variance of latent slopes (σ_S^2), respectively, with the sample size $N = 200$. The overall latent slope and variance of latent slopes are chosen as the parameters of interest, based on the presumption that substantive researchers using growth curve models are most often interested in assessing changes over time. Results for other model parameters and for $N = 500$ have similar patterns and are given in the supplementary file: `https://github.com/CynthiaXinTong/DOME`. When the sample size is 200, Geweke tests suggest that at least 91% replications converged. We summarize the estimation results based on those converged Markov chains.

When data contain outliers but no leverage observation exists, our proposed DOME growth curve modeling yields parameter estimates that are very similar to those from the conditional median-based method, and are less biased than those from traditional mean-based method. Both MSEs and standard errors from the two robust Bayesian methods are smaller than those from the traditional mean growth curve model, indicating that the proposed method is on par with conditional median-based method, and is more efficient than traditional growth curve modeling. This pattern is more salient when the proportion of outliers increases. Note that for DOME growth curve modeling, standard errors of σ_S^2 are underestimated, as ASEs are smaller than ESEs. This may be due to the autocorrelations of the Markov chains, and could potentially be overcome by thinning the Markov chains. In sum, the DOME growth curve modeling is more robust against outliers than traditional mean-based growth curve modeling. It provides less biased and more efficient parameter estimators. The conditional median-based method performs similarly as the DOME growth curve modeling on handling outliers.

When data contain leverage observations but no outliers, the advantage of the DOME GCM becomes apparent. In this situation, both traditional mean-based method and the conditional median-based growth curve modeling break down, yielding similar parameter estimates. But the estimates by DOME GCM are much less biased. For example, as shown in Table 1, when the proportion of leverage observations is 10%, the estimation bias for β_S is -0.45 for both traditional mean-based method and the robust median-based method. DOME growth curve modeling can substantially reduce the bias to -0.06. This is mainly because

Table 1. Estimation results for β_S under different types of data contamination when N = 200

Type	r%	Method	Est	Bias	MSE	ASE	ESE	CR	MET
Outlier	10%	Mean	1.33	-0.17	29.5	4.62	3.07	NA	
		Median	1.48	-0.02	1.21	2.50	2.42	94.2	1149
		DOME	1.48	-0.02	1.28	2.77	2.65	96	1551
	25%	Mean	1.06	-0.44	194.6	6.30	3.62	NA	
		Median	1.44	-0.06	4.73	3.19	2.78	94.4	740
		DOME	1.43	-0.07	5.17	3.35	3.05	95.8	1362
Leverage	10%	Mean	1.05	-0.45	202	9.82	2.26	NA	
		Median	1.05	-0.45	204	8.39	2.64	94.6	2583
		DOME	1.44	-0.06	5.32	4.21	3.17	93.4	2412
	25%	Mean	0.37	-1.13	1269	14.0	2.35	NA	
		Median	0.37	-1.13	1270	10.8	3.02	95.6	2963
		DOME	1.29	-0.21	47.1	7.35	4.09	94.6	2674
50-50 Mix	10%	Mean	1.19	-0.31	98.0	7.80	4.27	NA	
		Median	1.21	-0.29	86.2	7.05	5.40	95.0	2200
		DOME	1.45	-0.05	3.00	3.65	2.92	97.0	2072
	25%	Mean	0.72	-0.78	611	11.1	5.67	NA	
		Median	0.77	-0.73	545	9.69	7.56	94.6	1819
		DOME	1.36	-0.14	21.3	2.31	3.37	93.6	1936

Note. Est = Estimate; CR = convergence rate; MET = median estimation time in seconds; 50-50 Mix: data contain outliers and leverage observations with equal probabilities. MSE was multiplied by 1000 and ASE and ESE were multiplied by 100.

Table 2. Estimation results for σ_S^2 under different types of data contamination when N = 200

Type	r%	Method	Est	Bias	MSE	ASE	ESE	CR	MET
Outlier	10%	Mean	0.25	0.15	26.36	4.71	5.60	NA	
		Median	0.07	-0.03	1.02	1.31	1.26	93.8	982
		DOME	0.16	0.06	8.98	3.21	6.92	95.4	1263
	25%	Mean	0.39	0.29	88.57	8.74	7.04	NA	
		Median	0.06	-0.04	1.66	1.53	1.07	94.8	619.28
		DOME	0.13	0.03	1.51	4.68	11.97	93.6	1154
Leverage	10%	Mean	1.92	1.82	3326	19.32	6.33	NA	
		Median	1.93	1.83	3350	19.27	6.68	94.6	2253
		DOME	0.96	0.86	750	13.78	4.43	94.0	2087
	25%	Mean	3.90	3.80	14440	39.08	9.90	NA	
		Median	3.91	3.81	14503	38.74	11.46	94.8	2792
		DOME	3.64	3.55	12592	51.48	13.78	96.0	2279
50-50 Mix	10%	Mean	1.13	1.13	1098.89	12.30	20.06	NA	
		Median	0.11	0.01	10.32	34.41	10.13	91.0	2196
		DOME	0.18	0.08	41.08	57.74	18.41	93.0	2071
	25%	Mean	2.26	2.16	4737.37	216.05	26.35	NA	
		Median	0.11	0.01	15.65	1.05	12.47	92.6	1819
		DOME	0.51	0.41	277.74	41.60	32.35	94.4	1968

Note. Same as Table 1

the conditional median-based method only applies medians in the first level of the growth curve model, whereas leverage observations are extreme values in the second level. Note that standard errors in the DOME growth curve modeling is overestimated as ASEs are larger than ESEs. However, ASEs estimated by the DOME GCM method are still closer to their corresponding ESEs than the ASEs estimated by the other two methods.

When data contain both outliers and leverage observations, DOME growth curve modeling still performs much better than the mean-based method and the median-based method in terms of estimation bias and efficiency. In general, when data are suspected to be contaminated, DOME GCM should be a preferred method than the traditional GCM and conditional median-based GCM.

It is probably counter-intuitive that the MET is shorter when the proportion of outliers is higher. This is consistent with the findings in Tong et al. (2021). In MCMC sampling, Markov chains typically have trouble exploring high curvature regions. A small proportion of outliers (e.g., 10%) creates a steep and high curvature region for the chain to enter, and thus the computing time tends to be longer. As the proportion increases, the curvature becomes smoother and the MCMC procedure is faster.

4 A Real Data Application

To demonstrate its application, we apply the proposed DOME GCM to a subset of data from the Virginia Cognitive Aging Project (VCAP; Salthouse 2014, 2018). VCAP, starting in 2001, is currently one of the largest active longitudinal studies of aging involving comprehensive cognitive assessments in adults ranging from 18 to 99 years of age. Over 5,000 adults have participated in the three-session (6-8 hours) assessment at least once, with about 2,500 participating at least twice, and about 1300 participating three or more times. The subset we used contains observations on 338 participants, who made 5 visits to the assessment sessions. The change of memory scores over time is studied in this illustrative example. Traditional mean-based method and the conditional median-based method in Tong et al. (2021) are also applied to fit the dataset for comparison.

The trajectory plot for the memory scores (Figure 1) suggests a linear growth curve structure for the development of memory abilities. In the Bayesian estimation of DOME GCM, we assign a normal prior for the location vector $\boldsymbol{\beta}$, an inverse-gamma distribution for the measurement error variance σ_ϵ^2, and an inverse-Wishart for the random effect covariance Σ. For both the conditional median-based GCM and DOME GCM estimation, the total length of Markov chains is set as 15,000, with the first 7,500 draws being the burn-in period. Geweke statistics suggest that the Markov chains are stable after the burn-in period. The trace plots (Figures 2- 3) also suggest the convergence of the Markov chains.

The parameter estimates using the three methods are summarized in Table 3, and they differ a lot. The estimated rate of change $\hat{\beta}_S$ is 0.021 based on the

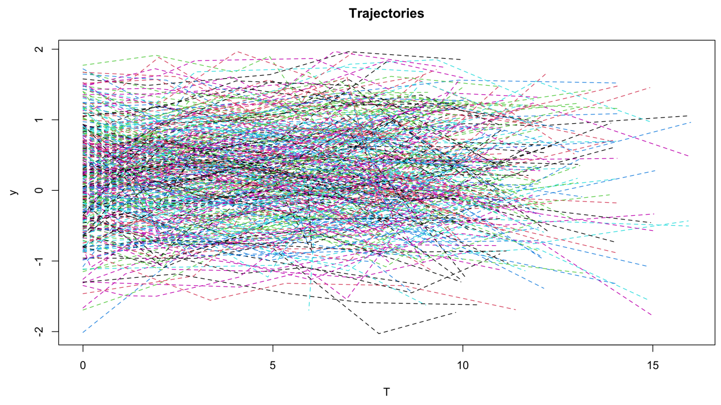


Figure 1. The trajectory plot for memory scores. Each lines is formed by connecting the consecutive measurements on the same individual.

traditional mean-based method. The estimates coming from the robust methods are much smaller, 0.003 from the conditional median-based approach, and 0.005 from the DOME growth curve modeling. Also, the 95% credible interval of $\hat{\beta}_S$ from traditional mean-based method is $[0.001, 0.042]$, suggesting the memory ability for the investigated population (median age of the group is 55) has a significant increasing trend. In contrast, the intervals produced by the two robust methods all cover zero, which is more reasonable and interpretable as most participants in the dataset are elderly. The parameter estimates from the two robust methods are similar. Based on our simulation results, when there is no leverage observation in the dataset, conditional median-based method and DOME growth curve modeling are expected to give similar results. That is most likely the case in this illustrative example. The difference between traditional method and the robust methods is the result of the presence of outliers in the dataset. As suggested in our simulation study, we should generally trust the results from DOME. The results from DOME show that the initial median memory ability is about 0.203. The credible intervals indicate that there are significant between-subject differences in both initial ability and the change over time. The covariance between the two random effects is -0.009, and is significantly different from 0, meaning that a higher initial level is associated with a slower rate of change in general.

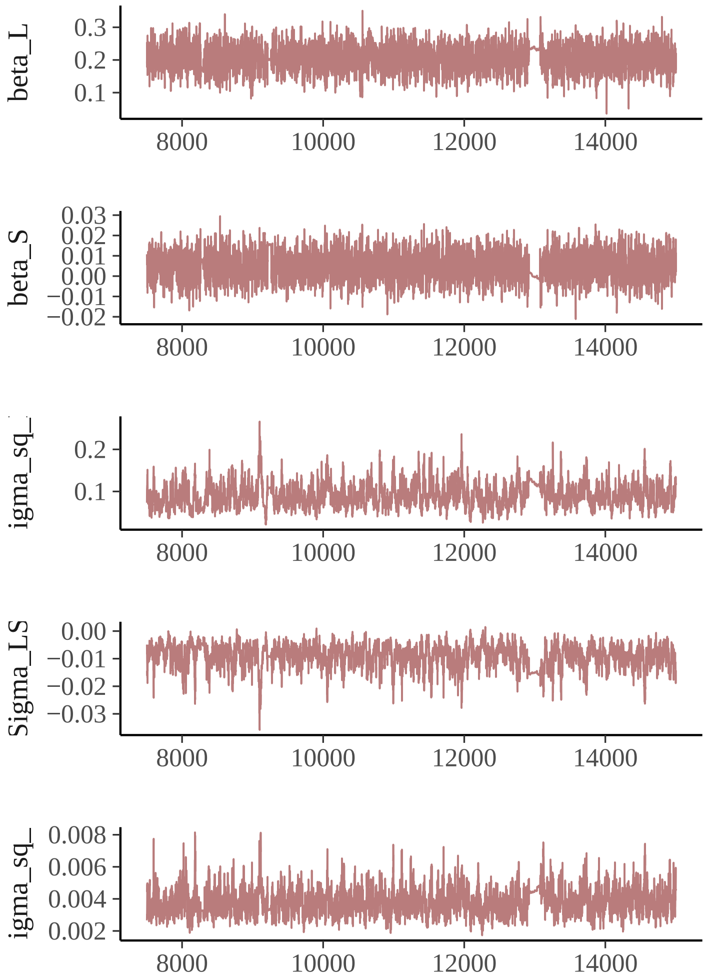


Figure 2. MCMC trace plot for the DOME growth curve modeling. Each line is formed by connecting consecutive draws of the same parameter. Only the draws after burn-in period of MCMC is used.

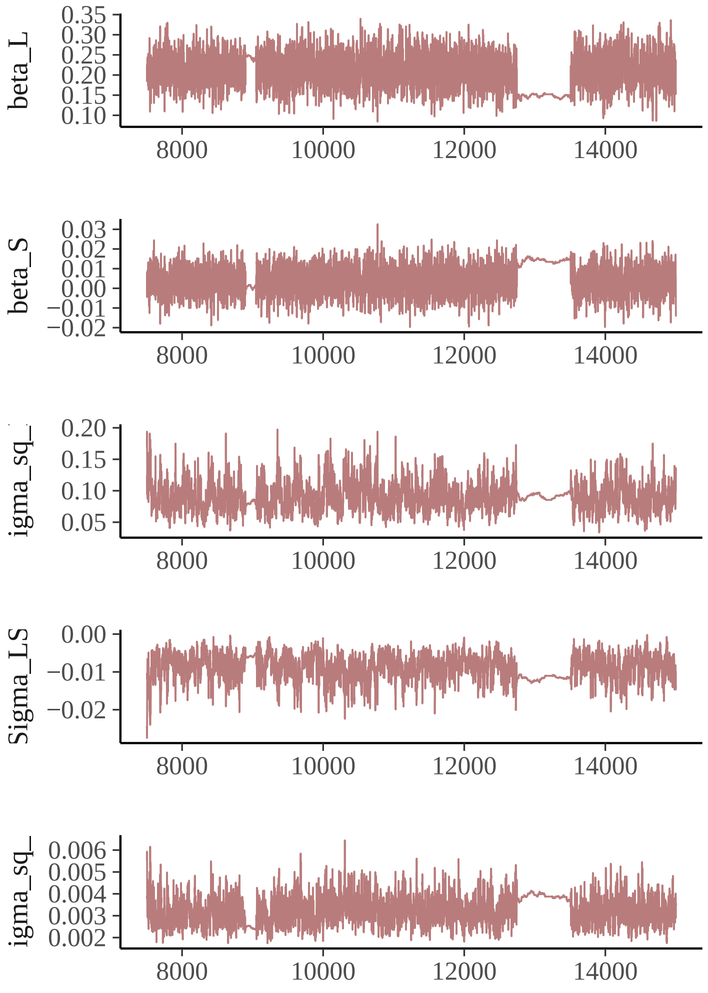


Figure 3. MCMC trace plot for the conditional median-based method. Each line is formed by connecting consecutive draws of the same parameter. Only the draws after burn-in period of MCMC is used.

Table 3. The estimates of memory ability real data application.

	Estimate	SE	CI	Geweke Statistic
	Mean GCM			
β_L	0.186	0.037	[0.114, 0.258]	
β_S	0.021	0.011	[0.001, 0.042]	
σ_L^2	0.374	0.036	[0.304, 0.444]	
σ_{LS}	-0.014	0.008	[-0.031, 0.002]	
σ_S^2	0.016	0.004	[0.001, 0.023]	
	Median GCM			
β_L	0.217	0.039	[0.139, 0.289]	-0.337
β_S	0.003	0.007	[-0.010, 0.017]	0.709
σ_L^2	0.089	0.023	[0.052, 0.140]	-0.707
σ_{LS}	-0.009	0.003	[-0.016, -0.003]	1.001
σ_S^2	0.003	0.001	[0.002, 0.005]	-0.603
	Double Median GCM			
β_L	0.203	0.038	[0.127, 0.278]	0.108
β_S	0.005	0.007	[-0.008, 0.018]	0.465
σ_L^2	0.093	0.028	[0.050, 0.158]	-0.790
σ_{LS}	-0.009	0.004	[-0.019, -0.003]	0.287
σ_S^2	0.004	0.001	[0.002, 0.006]	-0.573

5 Discussion

Growth curve modeling based on conditional medians has been developed to disentangle the influence of data contamination. In this paper, we developed a DOME GCM, a double medians based structure, to handle both outliers and leverage observations in longitudinal data. A simulation study was conducted to compare the numerical performances of traditional mean-based growth curve modeling, a median-based growth curve modeling, as well as the proposed DOME growth curve modeling. Results showed that when data were normally distributed, the three methods performed equally well. When data contain outliers but not leverage observations, the median-based method and DOME growth curve modeling yielded similar parameter estimates, which were less biased and more efficient than those from traditional growth curve modeling. When leverage observations existed, DOME growth curve modeling outperformed the other two approaches, providing much less biased parameter estimates. We therefore recommend to use DOME growth curve modeling in general as it can effectively handle both leverage observations and outliers.

As pointed out in Tong and Zhang (2017), outliers and leverage observations are equally likely to exist in samples in practice, but they affect model estimation differently. Although various methods have been developed to identify outliers and leverage observations separately, the accuracy and effectiveness of those methods were not guaranteed, especially in longitudinal studies. Tong and Zhang (2017) suggested that a final detection decision should rely on a combination of multiple methods. Our work in this paper indirectly provided an approach to imply whether data contain leverage observations or not, by comparing the estimation results from the median-based growth curve modeling and the DOME growth curve modeling. If their results greatly deviate from each other, we can conclude that leverage observations exist.

Note that estimations of the random effects parameters (e.g., σ_S^2) are not as good as those for the fixed effects (e.g., β_S) in general. This is consistent with the literature; namely, although the median has a higher breakdown point of 50%, it can be less efficient than the mean under some conditions. Thus, we need to carefully examine the estimated random effects parameters to determine whether there are significant between-subject variations in the within-subject change. One alternative approach is to extend the current approach based on conditional medians to approaches based on conditional quantiles. Such extension is natural with the assistance of asymmetric Laplace distributions, and we would be able to investigate the change pattern at different quantile levels inferring between-subject differences without investigating the random effects parameters.

In this study, we evaluated the performance of DOME growth curve modeling when data were contaminated. We want to point out that the data distribution nonnormality may be due to data contamination or nonnormal population distributions. Although we expect that the developed DOME GCM should still perform well when population distributions are nonnormal, it worths systematically assessing the effectiveness of DOME GCM and compare it with existing robust methods in future research.

Missing data in longitudinal data are inevitable, yet the conditional median based approaches have not been applied to analyze data with missing values. Since Bayesian methods were used for the DOME GCM estimation, multiple imputations can be automatically implemented and missing values can be accommodated relatively easily. We will extend the developed method to handle ignorable and non-ignorable missing data in future research.

Acknowledgment

This paper is based upon work supported by the National Science Foundation under grant no. SES-1951038.

References

Cho, H., Hong, H. G., & Kim, M.-O. (2016). Efficient quantile marginal regression for longitudinal data with dropouts. *Biostatistics*, *17*, 561–575. doi: https://doi.org/10.1093/biostatistics/kxw007

Depaoli, S., Liu, H., & Marvin, L. (2021). Parameter specification in bayesian cfa: An exploration of multivariate and separation strategy priors. *Structural Equation Modeling: A Multidisciplinary Journal*, *0*(0), 1-17. doi: https://doi.org/10.1080/10705511.2021.1894154

Galvao, A. F., & Poirier, A. (2019). Quantile regression random effects. *Annals of Econometrics and Statistics*, *134*, 109–148. (DOI: 10.15609/annaeconstat2009.134.0109) doi: https://doi.org/10.15609/annaeconstat2009.134.0109

Geraci, M. (2014). Linear quantile mixed models: the lqmm package for laplace quantile regression. *Journal of Statistical Software*, *57*, DOI: 10.18637/jss.v057.i13. doi: https://doi.org/10.18637/jss.v057.i13

Geweke, J. (1992). Evaluating the accuracy of sampling-based approaches to calculating posterior moments. In J. M. Bernado, J. O. Berger, A. P. Dawid, & A. F. M. Smith (Eds.), *Bayesian statistics 4* (pp. 169–193). Oxford, UK: Clarendon Press.

Huang, Y. (2016). Quantile regression-based Bayesian semiparametric mixed-effects modelsfor longitudinal data with non-normal, missing and mismeasured covariate. *Journal ofStatistical Computation and Simulation*, *86*, 1183–1202. (DOI: 10.1080/00949655.2015.1057732) doi: https://doi.org/10.1080/00949655.2015.1057732

Koenker, R. (2004). Quantile regression for longitudinal data. *Journal of Multivariate Data Analysis*, *91*, 74–89. doi: https://doi.org/10.1016/j.jmva.2004.05.006

Koenker, R., & Bassett, G. (1978). Regression quantiles. *Econometrica*, *46*, 33–50. doi: https://doi.org/10.2307/1913643

Kozubowski, T. J., & Podgorski, K. (2000). A multivariate and asymmetric generalization of laplace distribution. *Computational Statistics*, *15*, 531–540. (DOI: 10.1007/PL00022717) doi: https://doi.org/10.1007/pl00022717

Lange, K. L., Little, R. J. A., & Taylor, J. M. G. (1989). Robust statistical modeling uisng the t distribution. *Journal of the Americal Statistical Association*, *84*(408), 881–896. doi: https://doi.org/10.2307/2290063

Lu, Z., & Zhang, Z. (2014). Robust growth mixture models with non-ignorable missingness: Models, estimation, selection, and application. computational statistics and data analysis. *Computational Statistics and Data Analysis*, *71*, 220–240. doi: https://doi.org/10.1016/j.csda.2013.07.036

McArdle, J. J., & Nesselroade, J. R. (2014). *Longitudinal data analysis using structural equation models.* American Psychological Association. doi: https://doi.org/10.1037/14440-000

Micceri, T. (1989). The unicorn, the normal curve, and other improbable creatures. *Psychological Bulletin*, *105*(1), 156–166. doi: https://doi.org/10.1037/0033-2909.105.1.156

Pendergast, J. F., & Broffitt, J. D. (1985). Robust estimation in growth curve models. *Communications in Statistics: Theory and Methods*, *14*, 1919–1939. doi: https://doi.org/10.1080/03610928508829021

Reich, B. J., Bondell, H. D., & J., W. H. (2010). Flexible bayesian quantile regression for independent and clustered data. *Biostatistics*, *11*, 337–352. doi: https://doi.org/10.1093/biostatistics/kxp049

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

Salthouse, T. A. (2014). Correlates of cognitive change. *Jour-*

nal of Experimental Psychology: General, *143*, 1026–1048. doi: https://doi.org/10.1037/a0034847

Salthouse, T. A. (2018). Why is cognitive change more negative with increased age? *Neuropsychology*, *32*, 110–120. doi: https://doi.org/10.1037/neu0000397

Singer, J. M., & Sen, P. K. (1986). M-methods in growth curve analysis. *Journal of Statistical Planning and Inference*, *13*, 251–261. doi: https://doi.org/10.1016/0378-3758(86)90137-0

Smith, L. B., Fuentes, M., Gordon-Larsen, P., & Reich, B. J. (2015). Quantile regression for mixed models with an application to examine blood pressure trends in china. *The Annals of Applied Statistics*, *9*, 1226–1246. doi: https://doi.org/10.1214/15-aoas841

Stan Development Team. (2019). *RStan: the R interface to Stan.* Retrieved from `http://mc-stan.org/` (R package version 2.19.2)

Tong, X., Zhang, T., & Zhou, J. (2021). Robust bayesian growth curve modelling using conditional medians. *British Journal of Mathematical and Statistical Psychology*, *74*(2), 286-312. doi: https://doi.org/10.1111/bmsp.12216

Tong, X., & Zhang, Z. (2012). Diagnostics of robust growth curve modeling using student's t distribution. *Multivariate Behavioral Research*, *47*, 493–518. doi: https://doi.org/10.1080/00273171.2012.692614

Tong, X., & Zhang, Z. (2017). Outlying observation diagnostics in growth curve modeling. *Multivariate Behavioral Research*, *52*, 768–788. doi: https://doi.org/10.1080/00273171.2017.1374824

Yuan, K.-H., & Bentler, P. M. (1998a). Structural equation modeling with robust covariances. *Sociological Methodology*, *28*, 363–396. doi: https://doi.org/10.1111/0081-1750.00052

Yuan, K.-H., & Bentler, P. M. (2001). Effect of outliers on estimators and tests in covariance structure analysis. *British Journal of Mathematical and Statistical Psychology*, *54*, 161–175. doi: https://doi.org/10.1348/000711001159366

Zhang, H., Huang, Y., Wang, W., Chen, H., & Langland-Orban, B. (2019). Bayesian quantile regression-based partially linear mixed-effects joint models for longitudinal data with multiple features. *Statistical Methods in Medical Research*, *28*, 569–588. (DOI: 10.1177/0962280217730852) doi: https://doi.org/10.1177/0962280217730852

Zhong, X., & Yuan, K.-H. (2010). Weights. In N. J. Salkind (Ed.), *Encyclopedia of research design* (pp. 1617–1620). Thousand Oaks, CA: Sage.

Zhong, X., & Yuan, K.-H. (2011). Bias and efficiency in structural equation modeling: Maximum likelihood versus robust methods. *Multivariate Behavioral Research*, *46*, 229–265. doi: https://doi.org/10.1080/00273171.2011.558736

Appendix: Implementation

The “rstan” package (Stan Development Team, 2019) is used in our study. Below we provide the annotated R code for the real data analysis.

```
DOME<-"
data{
  int<lower=0> N;
  int<lower=0> T;
  vector[N*T] X;
  vector[N*T] y;

  // fixed inv_gamma parameter, prior of epsilon variance
  real shape;
  real inv_scale;

  // beta prior information, the global slope&intercept
  vector[2] beta_0;
  cov_matrix[2] Var_beta0;

  // hyper parameter’s value for Var_b
  cov_matrix[2] Var_b0;
}

transformed data{
  int len;
  len = N*T;
}

parameters{
  real<lower=0> sigma; //epsilon variance
  vector<lower=0>[len] v; // data augmentation, represent
                          // Laplace epsilon in expo
  vector<lower=0>[N] vb; // data augmentation, MLD b

  vector[2] beta;
  vector[2] b_star[N];

  cov_matrix[2] Var_b;
  cov_matrix[2] Var_beta;
}

transformed parameters{
  vector[len] mu;
  vector[len] sigma_y;
  vector[N] vb_root;
  vb_root = sqrt(vb);

  for(i in 1:N){
    for(j in 1:T){
      mu[T*(i-1)+j] = beta[1] + b_star[i,1]*vb_root[i] +
        (beta[2] + b_star[i, 2]*vb_root[i]) * X[T*(i-1)+j];
    }
  }

  sigma_y = sqrt(sigma*v);
}

model{
  // model
  y ~ normal(mu, sigma_y);

  // data augmentation
  sigma ~ inv_gamma(shape, inv_scale);
  v ~ exponential(1);
  vb ~ exponential(1);

  // priors
  beta ~ multi_normal(beta_0, Var_beta);
  // b_star * sqrt(vb) is b, written this way to vectorize
  b_star ~ multi_normal([0, 0], Var_b);
  Var_beta ~ inv_wishart(3, Var_beta0);
  Var_b ~ inv_wishart(3, Var_b0);
}
"
#load VCAP data and prepare intial values for MCMC
y<-as.vector(y)
X<-as.vector(time)
N<-length(y)/5
lm_est<-lm(y~X)
beta_0<-c(lm_est$coefficients[1],lm_est$coefficients[2])
Var_beta0<-matrix(c(0.5,0,0,0.1),ncol = 2)
Var_b0<-matrix(c(0.5,0,0,0.1),ncol = 2)
dat<-list(N=N, T=T, y=y, X=X, beta_0=beta_0, shape=0.1,
   inv_scale=0.1, Var_beta0=Var_beta0, Var_b0=Var_b0)
v_initial<-rep(1,N*T)
vb_initial<-rep(1,N)
b_initial<-matrix(rep(0, N*2), N, 2)
intial<-list(list(sigma=runif(1,0.5,2), beta=beta_0,
    b=b_initial,  v=v_initial, vb=vb_initial,
    Var_beta=Var_beta0, Var_b=Var_b0))

#fit the DOME model using rstan package
fit_DOME<-stan(model_code=double_quantile, model_name="DOME",
  init=intial, pars=c("beta","Var_b"), data=dat, iter=15000,
  chains=1)
summary(fit_DOME)$summary

//check convergence via geweke test
content<-extract(fit_double_median)
geweke.diag(content$beta[,1])$z
geweke.diag(content$beta[,2])$z
geweke.diag(content$Var_b[, 1, 1])$z
geweke.diag(content$Var_b[, 1, 2])$z
geweke.diag(content$Var_b[, 2, 2])$z

#draw traceplot for MCMC, serving as reference for convergence
color_scheme_set('mix-blue-red')
mcmc_trace(fit_DOME,
    pars = c("beta[1]","beta[2]", "Var_b[1,1]", "Var_b[1,2]",
    "Var_b[2,2]"), facet_args = list(ncol = 1,
    strip.position = "left"), iter1 = 7500)
```

Journal of Behavioral Data Science, 2022, *2* (2), 23–46.
DOI: `https://doi.org/10.35566/jbds/v2n2/p2`

A New Bayesian Structural Equation Modeling Approach with Priors on the Covariance Matrix Parameter

Haiyan Liu[1], Wen Qu[2], Zhiyong Zhang[3], and Hao Wu[4]

[1] University of California-Merced, Merced, CA 95343, USA
`hliu62@ucmerced.edu`
[2] Fudan Institute for Advanced Study in Social Sciences, Fudan Univeristy, Shanghai, China
`wqu@fudan.edu.cn`
[3] University of Notre Dame, Notre Dame, IN 46530, USA
`zzhang4@nd.edu`
[4] Vanderbilt University, Nashville, TN 37203, USA
`hao.wu.1@Vanderbilt.edu`

Abstract. Bayesian inference for structural equation models (SEMs) is increasingly popular in social and psychological sciences owing to its flexibility to adapt to more complex models and the ability to include prior information if available. However, there are two major hurdles in using the traditional Bayesian SEM in practice: (1) the information nested in the prior distributions is hard to control, and (2) the MCMC iterative procedures naturally lead to Markov chains with serial dependence and the diagnostics of their convergence are often difficult. In this study, we present an alternative procedure for Bayesian SEM aiming to address the two challenges. In the new Bayesian SEM procedure, we specify a prior distribution on the population covariance matrix parameter $\mathbf{\Sigma}$ and obtain its posterior distribution $p(\mathbf{\Sigma}|\text{data})$. We then construct a posterior distribution of model parameters $\boldsymbol{\theta}$ in the hypothetical SEM model by transforming the posterior distribution of $\mathbf{\Sigma}$ to a distribution of model parameter $\boldsymbol{\theta}$. The new procedure eases the practice of Bayesian SEM significantly and has a better control over the information nested in the prior distribution. We evaluated its performance through a simulation study and demonstrate its application through an empirical example.

Keywords: Structural equation modeling · Bayesian analysis · Inverse Wishart prior · Informative prior · Convergence diagnostics

1 Introduction

Structural equation modeling (SEM) is widely used to analyze multivariate data with complex structures in behavioral and social sciences, due to its ability to identify relationships among unobserved latent variables using the observed data (e.g., P. Bentler & Dudgeon, 1996; Bollen, 1989; Jöreskog, 1978; Lee, 2007; MacCallum & Austin, 2000). In a typical SEM model, manifest variables, latent variables, and measurement error can be analyzed simultaneously (e.g., Anderson & Gerbing, 1988; Yuan, Kouros, & Kelley, 2008). The family of SEMs contains a large variety of well-known models such as path analysis models (e.g., Boker & McArdle, 2005; Yuan et al., 2008), confirmatory factor models (e.g., Jöreskog, 1969), and growth curve models (e.g., Grimm, Steele, Ram, & Nesselroade, 2013; McArdle & Nesselroade, 2003). One primary purpose of fitting an SEM model is to explain the covariance structure among variables, either latent or manifest. Such a covariance structure is depicted by parameters in the hypothetical model such as factor loadings, path coefficients, factor covariance matrices, and residual variances.

Bayesian methods are increasingly used in estimating SEMs (e.g. Lee, 2007; Palomo, Dunson, & Bollen, 2007; Van de Schoot, Winter, Ryan, Zondervan-Zwijnenburg, & Depaoli, 2017). The seminal work by Lee (2007) laid the ground for Bayesian SEM. Guo, Zhu, Chow, and Ibrahim (2012) used Bayesian Lasso for model regularization. Zhang, Lai, Lu, and Tong (2013) introduced a robust Bayesian method to estimate growth curve models. Wang, Feng, and Song (2016) employed the Bayesian method in estimating quantile SEMs. Muthen and Asparouhov (2012) proposed to use small variance priors to approximate parameters typically specified to be 0 in the frequentist framework. The increasing popularity of Bayesian methods first benefits from the computer hardware development that renders sampling techniques such as Markov Chain Monte Carlo (MCMC) samplers (Gelfand & Smith, 1990). It is also because of the many beneficial features of Bayesian statistics (Van de Schoot et al., 2017). For example, it is possible to incorporate prior information into the estimation process in data analysis, which has the analogous contribution of extra data and is particularly useful when the sample size is small where the maximum likelihood (ML) method meets troubles to converge (e.g., P. M. Bentler & Yuan, 1999). It can also be computationally tractable even for very complex models (e.g., Muthen & Asparouhov, 2012). In addition, it is especially flexible to handle missing data and latent variables using data augmentation techniques (e.g., Z. Lu, Zhang, & Lubke, 2011; Van Dyk & Meng, 2001; Zhang & Wang, 2012). Moreover, it treats model parameters as random variables with a more intuitive interpretation (Van de Schoot et al., 2017). Because of the increasing popularity of Bayesian statistical inference, the software has also been developed to facilitate the use of Bayesian SEM such as the R package *blavaan* (Merkle & Rosseel, 2015) and *Mplus* (Muthen & Asparouhov, 2012).

Despite the many advantages, the burden of using Bayesian SEM is also discussed (e.g., MacCallum, Edwards, & Cai, 2012). A common criticism on Bayesian statistics is the use of priors. First, it can be difficult to specify priors and risky to use default priors in software (Liu, Depaoli, & Marvin, 2022; Smid, McNeish, Miočević, & van de Schoot, 2019). In an SEM, there are many parameters including factor loadings, factor covariance matrix, unique factor variances, regression paths, and other parameters. Selecting priors for all those parameters can be tedious. Although default priors are provided in most existing software, their influences are still not well understood by many applied researchers, especially for complex models and/or small sample size studies (Depaoli, Liu, & Marvin, 2021; Smid et al., 2019). Second, the use of informative priors can significantly affect parameter estimates and Bayesian inference. In order to reduce the influence of priors and obtain objective inference, Jeffreys prior has been developed (Jeffreys, 1961). However, the derivation of Jeffreys prior is not straightforward and therefore convenient priors such as univariate and multivariate normal, Gamma, and Wishart priors are often used in practice (e.g., Lee & Song, 2012; Song & Lu, 2010; Zhang, Hamagami, Lijuan Wang, Nesselroade, & Grimm, 2007; Zhang et al., 2013). Third, some researchers have argued that informative priors should be used to fully take advantage of Bayesian inference (e.g., Z.-H. Lu, Chow, & Loken, 2016; Muthen & Asparouhov, 2012). However, the specification of such priors is even more delicate. Another concern is the use of Markov chain Monte Carlo (MCMC) techniques. The convergence diagnostics of the Markov chains are required and can be very challenging. The dependence among the Markov samples often suggests a long chain to provide reliable inferences. Therefore, the specification of priors and the diagnostics of convergence have become two major obstacles for the adoption of Bayesian SEM in practice. Although some existing software has implemented Bayesian MCMC techniques, it might not work well with the default settings for all models.

The objective of this paper is to present an alternative Bayesian approach to SEM that specifies prior distributions under a unified framework for all covariance structures. Rather than specifying priors on individual model parameters $\boldsymbol{\theta}$, this approach specifies a multivariate prior distribution directly on the saturated population covariance matrix $\mathbf{\Sigma}$. Random draws from the posterior distribution of the population covariance matrix are then fitted to the covariance structure to obtain posterior distributions of model parameters.

Compared to the existing procedures for Bayesian SEM, the newly proposed procedure has two distinct features. First, priors are only required for the population covariance matrix and there is no need to specify a prior for each individual model parameter such as factor loadings and residual variances. This substantially reduces the work for researchers, especially novel users of Bayesian methods, conducting a Bayesian SEM analysis. Inverse-Wishart priors have been adopted for the covariance matrix parameter in most of the existing Bayesian analyses (e.g., Grimm, Kuhl, & Zhang, 2013; Liu, Zhang, & Grimm, 2016; Pan, Song, Lee, & Kwok, 2008; Zhang, 2021; Zhang et al., 2007, 2013). In the current study, we also use Inverse-Wishart priors becasue it is not only computationally convenient but also practically meaningful. The prior information conveyed by an Inverse Wishart prior $\mathrm{IW}(m, \mathbf{V})$ is the same as "additional" data with the sample size m and the sum of squares $\mathbf{V}$. As a result, we are able to control the amount of prior information nested in the prior by adjusting the values of hyperparameters of an Inverse-Wishart distribution. Second, the posterior samples of model parameters $\boldsymbol{\theta}$ are independently and identically distributed. This is because the posterior distribution of the population covariance matrix is a marginal distribution and has a closed form. Thus, independent covariance matrices can be sampled from the posterior distribution directly. And so are the $\boldsymbol{\theta}'\mathbf{s}$ mapped from them. Because every obtained sample is from its posterior distribution, it is not necessary to have a burn-in period and conduct convergence diagnostics. Because of the independence of samples, relatively fewer samples are needed to obtain reliable inferences compared to the conventional MCMC procedures.

The rest of this paper is organized as follows. We first provide a brief introduction to both structural equation modeling and Bayesian statistical inference.We then present our new approach to Bayesian SEM. The use of prior distributions, types of points estimates, and credible intervals are described. After that, we evaluate the performance and illustrate the application of the new procedure using a confirmatory factor model. Finally, we conclude the paper with discussions and future directions.

2 Introduction to SEM and Bayesian Inference

In this section, we provide a brief review of SEM and Bayesian inference in the aim to explain some notations used in the rest of the paper. More details can be found in the seminal works such as Gelman et al. (2013), Kruschke (2011), Lee (2007) and Lee and Song (2012).

2.1 SEM

A classical SEM contains two components: a measurement model and a structure model. Using the LISREL (Jöreskog & Sörbom, 1993) notations, it can be represented as follows,

$$\begin{aligned} \mathbf{x} &= \boldsymbol{\Lambda} \begin{bmatrix} \boldsymbol{\eta} \\ \boldsymbol{\xi} \end{bmatrix} + \varepsilon \\ \boldsymbol{\eta} &= \mathbf{B}\boldsymbol{\eta} + \boldsymbol{\Gamma}\boldsymbol{\xi} + \boldsymbol{\delta} \end{aligned} \tag{1}$$

where $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ are latent dependent (endogenous) and independent (exogenous) variables; $\mathbf{x}$ is a vector of their indicators; $\boldsymbol{\Lambda}$ is a factor loading matrix; $\mathbf{B}$ and $\boldsymbol{\Gamma}$ are two coefficient matrices to represent the relationship among latent dependent variables and between latent independent and latent dependent variables, respectively; and $\boldsymbol{\epsilon}$ a vector of unique factors and $\boldsymbol{\delta}$ represents the residuals of the structure model. The elements of $\boldsymbol{\varepsilon}$ are independent of the elements in $\boldsymbol{\delta}$, but the elements in $\boldsymbol{\delta}$ are allowed to be correlated with each other as in the original LISREL model. For convenience, let $\boldsymbol{\Psi}$ be the covariance matrix of $\boldsymbol{\varepsilon}$, $\boldsymbol{\Phi}$ be the covariance matrix of various latent independent variables so that $\mathrm{cov}(\boldsymbol{\xi}) = \boldsymbol{\Phi}$, and $\boldsymbol{\Theta} = \mathrm{cov}(\boldsymbol{\delta})$. We denote all the unknown parameters as $\boldsymbol{\theta}$ that consists of factor loadings $\boldsymbol{\Lambda}$, path coefficients $\mathbf{B}$ and $\boldsymbol{\Gamma}$, as well as $\boldsymbol{\Phi}$ and $\boldsymbol{\Theta}$. The model implied covariance matrix $\Sigma(\boldsymbol{\theta})$ is a function of the unknown parameters $\boldsymbol{\theta}$. In order to estimate the parameter $\boldsymbol{\theta}$, both maximum likelihood (ML) estimation and Bayesian estimation methods can be used.

2.2 Maximum likelihood estimation of SEM

Following the tradition in structural equation modeling, the observed variable X is assumed to follow a multivariate normal distribution. When the covariance structure is of primary interests, the following discrepancy function is usually minimized to obtain the maximum likelihood estimates of $\boldsymbol{\theta}$

$$F_{ML} = \log|\mathbf{\Sigma}(\boldsymbol{\theta})| + \mathrm{tr}(\mathbf{S}\mathbf{\Sigma}^{-1}(\boldsymbol{\theta})) - \log|\mathbf{S}| - p \tag{2}$$

where p is the number of manifest variables and $\mathbf{S}$ is the sample covariance matrix as defined previously. Newton-types of algorithms are often employed to get ML estimates, which we refer to as $\hat{\boldsymbol{\theta}}_{ML}$ in this study.

2.3 Bayesian estimation of SEM

In Bayesian statistical inference, the model parameter $\boldsymbol{\theta}$ is not a fixed number, but a random variable following a probability distribution. The joint distribution of data $\mathbf{X}$ and model parameters $\boldsymbol{\theta}$ is

$$P(\mathbf{X}, \boldsymbol{\theta}) = P(\mathbf{X}|\boldsymbol{\theta})P(\boldsymbol{\theta}),$$

from which we can obtain the posterior distribution of $\boldsymbol{\theta}$ conditional on n independent observations on $\mathbf{X} : \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n$,

$$P(\boldsymbol{\theta}|\mathbf{x}_1, \cdots, \mathbf{x}_n) = \frac{\Pi_{i=1}^n P(\mathbf{x}_i|\boldsymbol{\theta})P(\boldsymbol{\theta})}{\Pi_{i=1}^n \int_{\boldsymbol{\theta}} P(\mathbf{x}_i, \boldsymbol{\theta}) d\boldsymbol{\theta}}. \tag{3}$$

Here, $P(\boldsymbol{\theta}|\mathbf{x}_1, \cdots, \mathbf{x}_n)$ is called the posterior distribution of $\boldsymbol{\theta}$ given data, and $\Pi_{i=1}^n P(\mathbf{x}_i|\boldsymbol{\theta})$ is the likelihood function, which is a function of $\boldsymbol{\theta}$. The term $P(\boldsymbol{\theta})$ represents the prior distribution of $\boldsymbol{\theta}$, which summarizes the information on the model parameters based on the prior knowledge. Because the denominator of the above equation is the marginal distribution of data and it is a constant with respect to model parameters $\boldsymbol{\theta}$, we use $P(\mathbf{X})$ to represent it for convenience. The posterior distribution is then

$$\begin{aligned} P(\boldsymbol{\theta}|\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n) &= \frac{1}{P(\mathbf{X})} \Pi_{i=1}^n P(\mathbf{x}_i|\boldsymbol{\theta})P(\boldsymbol{\theta}) \\ &\propto \Pi_{i=1}^n P(\mathbf{x}_i|\boldsymbol{\theta})P(\boldsymbol{\theta}) \end{aligned} \tag{4}$$

where the symbol $\propto$ means that a constant for scaling is removed. The posterior distribution itself is the combination of the likelihood function and the prior.

In SEM, there are latent variables involved. To obtain Bayesian inference, the data augmentation technique is usually adopted (Rubin, 1987; Tanner & Wong, 1987). Instead of working on the posterior marginal distribution of the unknown parameters, one could choose to work on the joint posterior distribution of unknown parameters and latent variables. For instance in a LISREL model, the joint posterior distribution is as follows

$$\begin{aligned}
&P(\boldsymbol{\Lambda}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \sigma_k^2, \boldsymbol{\eta}, \boldsymbol{\xi} | \mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n) \\
&\qquad \propto P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n | \boldsymbol{\Lambda}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \sigma_k^2, \boldsymbol{\eta}, \boldsymbol{\xi}) \\
&\qquad \quad \times P(\boldsymbol{\Lambda}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \sigma_k^2, \boldsymbol{\eta}, \boldsymbol{\xi}) \\
&\qquad = P(\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n | \boldsymbol{\Lambda}, \sigma_k^2, \boldsymbol{\eta}, \boldsymbol{\xi}) P(\boldsymbol{\eta} | \boldsymbol{\xi}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Theta}) \\
&\qquad \quad \times P(\boldsymbol{\xi} | \boldsymbol{\Phi}) P(\boldsymbol{\Lambda}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \sigma_k^2, k = 1, \cdots, p)
\end{aligned}$$

with $P(\boldsymbol{\Lambda}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \sigma_k^2)$ being the joint prior distribution of unknown parameters. The adoption of such a technique does not only make the inference on model parameters easier analytically, but also provides Bayesian inference on latent variables directly.

In most existing Bayesian SEM studies, independent priors are used (e.g., Lee, 2007; Muthen & Asparouhov, 2012; Zhang et al., 2013) such that

$$P(\boldsymbol{\Lambda}, \mathbf{B}, \boldsymbol{\Gamma}, \boldsymbol{\Phi}, \boldsymbol{\Theta}, \sigma_k^2) = P(\boldsymbol{\Lambda}) P(\mathbf{B}) P(\boldsymbol{\Gamma}) P(\boldsymbol{\Phi}) P(\boldsymbol{\Theta}) P(\sigma_k^2, k = 1, \cdots, p) \tag{5}$$

For example, the following are the types of priors used as the default priors in the existing software such as *Mplus* and *blavaan*,

$$\begin{aligned}
\sigma_k^2 &\sim \text{Inverse Gamma}(\alpha_{0k}, \beta_{0k}) \\
\boldsymbol{\Lambda}_k &\sim \text{N}(\Lambda_{0k}, H_{0k}) \\
\mathbf{B}_k &\sim \text{N}(B_{0k}, J_{0k}) \\
\boldsymbol{\Gamma}_k &\sim \text{N}(\gamma_{0k}, K_{0k}) \\
\boldsymbol{\Phi} &\sim \text{IW}(T_0, \beta_0) \\
\boldsymbol{\Theta} &\sim \text{IW}(R_0, \rho_0)
\end{aligned}$$

in which σ_k^2 is the error variance of the kth observed variable;$\boldsymbol{\Lambda}_k, \mathbf{B}_k, \boldsymbol{\Gamma}_k$ are the kth row of the factor loadings and path coefficients matrices, respectively; and $\alpha_{0k}, \beta_{0k}, \Lambda_{0k}, H_{0k}, B_{0k}, J_{0k}, \gamma_{0k}, K_{0k}, T_0, \beta_0, R_0$, and ρ_0 are the hyperparameters of the prior distributions, whose values are designated based on the prior information.

To obtain samples from the posterior distribution, Markov Chain Monte Carlo (MCMC) iterative procedures such as Gibbs samplers are used in existing Bayesian SEM. With the given starting values $\boldsymbol{\Lambda}^0, \mathbf{B}^0, \boldsymbol{\Gamma}^0, \boldsymbol{\Phi}^0, \boldsymbol{\Theta}^0, (\sigma_k^2)^0$,$(\boldsymbol{\eta}^0, \boldsymbol{\xi}^0)$, at jth iteration, sample

1. $\boldsymbol{\Phi}^j$ from $P(\boldsymbol{\Phi} | \boldsymbol{\xi}^{j-1}, \boldsymbol{\Lambda}^{j-1}, (\sigma_k^2)^{j-1}, \boldsymbol{\eta}^{j-1}, \mathbf{B}^{j-1}, \boldsymbol{\Gamma}^{j-1}, \boldsymbol{\Theta}^{j-1}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$
2. $\boldsymbol{\xi}^j$ from $P(\boldsymbol{\xi} | \boldsymbol{\Phi}^j, \boldsymbol{\Lambda}^{j-1}, (\sigma_k^2)^{j-1}, \boldsymbol{\eta}^{j-1}, \mathbf{B}^{j-1}, \boldsymbol{\Gamma}^{j-1}, \boldsymbol{\Theta}^{j-1}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$
3. $\boldsymbol{\eta}^j$ from $P(\boldsymbol{\eta} | \boldsymbol{\Phi}^j, \boldsymbol{\Lambda}^{j-1}, (\sigma_k^2)^{j-1}, \boldsymbol{\xi}^j, \mathbf{B}^{j-1}, \boldsymbol{\Gamma}^{j-1}, \boldsymbol{\Theta}^{j-1}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$
4. $\mathbf{B}^j$ from $P(\mathbf{B} | \boldsymbol{\Phi}^j, \boldsymbol{\Lambda}^{j-1}, (\sigma_k^2)^{j-1}, \boldsymbol{\eta}^j, \boldsymbol{\xi}^j, \boldsymbol{\Gamma}^{j-1}, \boldsymbol{\Theta}^{j-1}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$
5. $\boldsymbol{\Gamma}^j$ from $P(\boldsymbol{\Gamma} | \boldsymbol{\Phi}^j, \boldsymbol{\Lambda}^{j-1}, (\sigma_k^2)^{j-1}, \boldsymbol{\eta}^j, \boldsymbol{\xi}^j, \mathbf{B}^j, \boldsymbol{\Theta}^{j-1}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$
6. $\boldsymbol{\Lambda}^j$ from $P(\boldsymbol{\Lambda} | \boldsymbol{\Phi}^j, \boldsymbol{\Gamma}^j, (\sigma_k^2)^{j-1}, \boldsymbol{\eta}^j, \boldsymbol{\xi}^j, \mathbf{B}^j, \boldsymbol{\Theta}^{j-1}, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$
7. $\boldsymbol{\Theta}^j$ from $P(\boldsymbol{\Theta} | \boldsymbol{\Phi}^j, \boldsymbol{\Gamma}^j, (\sigma_k^2)^{j-1}, \boldsymbol{\eta}^j, \boldsymbol{\xi}^j, \mathbf{B}^j, \boldsymbol{\Lambda}^j, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$

8. $(\sigma_k^2)^j$ from $P(\sigma_k^2|\boldsymbol{\Phi}^j, \boldsymbol{\Gamma}^j, \boldsymbol{\Lambda}^j, \boldsymbol{\eta}^j, \boldsymbol{\xi}^j, \mathbf{B}^j, \boldsymbol{\Theta}^j, \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_n)$, for $k = 1, 2, \cdots, p$.

In SEM, the above conditional posterior distributions very often do not have analytically closed forms. Obtaining samples from the conditional distributions can be very complex and sampling schemes, such as the Metropolis–Hastings algorithms, are usually used within each step to draw samples from the conditional posterior distributions. Let $\boldsymbol{\theta}$ be the vector of all model parameters; by repeating the above Gibbs procedure B times, where B is usually a large number, we will obtain a chain of posterior samples $\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, \cdots, \boldsymbol{\theta}^t, \cdots, \boldsymbol{\theta}^B$.

Convergence diagnostics of the Markov chains are required in Bayesian analysis (e.g., Brooks & Roberts, 1998). This is because only the part of chains that has reached the stationary status would be representative of the posterior distribution. However, the diagnostics logically cannot guarantee representativeness. In addition, the successive draws by MCMC are correlated. Although the existence of auto-correlations does not necessarily mean bad point estimates, but correlated samples provide much less information than the same amount of uncorrelated ones. Higher auto-correlations usually suggest that longer chains are needed to have reliable inferences.

To summarize, in conventional Bayesian SEM, priors are specified on individual unknown parameters. MCMC procedures are used to obtain samples from the posterior distributions of model parameters. Convergence diagnostics of the Markov chains are required. Athough these can be done by software through default settings, it might lead to serious problems if researchers are not familiar with Bayesian methods. Hence, it is still not easy to conduct Bayesian SEM in general.

3 Proposed Bayeian SEM Approach: Prior on the Covariance Matrix Parameter

3.1 The general framework

In this work we propose a different framework for Bayesian SEM. In this approach, a prior distribution is imposed on the space of saturated covariance matrices:

$$\boldsymbol{\Sigma} \sim \pi(\boldsymbol{\Sigma})$$

The choice of prior distribution π will be discussed shortly. As usual, the observations are assumed to independently follow a multivariate normal distribution. This implies

$$\mathbf{S}|\boldsymbol{\Sigma} \sim \mathrm{W}(\boldsymbol{\Sigma}/(n-1), (n-1))$$

where n is sample size and $\mathbf{S}$ is the usual unbiased estimator of $\boldsymbol{\Sigma}$.

It should be noted that the parameters in the covariance structure are not explicit in the formulation above; rather, they are considered implicit functions of variances and covariances in $\boldsymbol{\Sigma}$:

$$\boldsymbol{\theta}^* = \mathbf{argmin}\ F_{ML}(\boldsymbol{\Sigma}, \boldsymbol{\Sigma}(\boldsymbol{\theta}))$$

In addition, the discrepancy between the true population covariance matrix $\mathbf{\Sigma}$ and the model implied covariance matrix $\mathbf{\Sigma}^* = \mathbf{\Sigma}(\boldsymbol{\theta}^*)$ is also a function of $\mathbf{\Sigma}$:

$$F^* = \min_{\theta} F_{ML}(\mathbf{\Sigma}, \mathbf{\Sigma}(\boldsymbol{\theta})) = F_{ML}(\mathbf{\Sigma}, \mathbf{\Sigma}(\boldsymbol{\theta}^*))$$

3.2 Prior and posterior distributions of Σ

The Wishart likelihood function of $\mathbf{\Sigma}$ is given by

$$L(\mathbf{\Sigma}|\mathbf{S}) \propto |\mathbf{\Sigma}|^{-n/2} \exp[-\frac{1}{2}\text{tr}(n\mathbf{S}\mathbf{\Sigma}^{-1})]$$

The posterior distribution is therefore

$$\pi(\mathbf{\Sigma}|\mathbf{S}) \propto |\mathbf{\Sigma}|^{-n/2} \exp[-\frac{1}{2}\text{tr}(n\mathbf{S}\mathbf{\Sigma}^{-1})]\pi(\mathbf{\Sigma}). \tag{6}$$

Many different priors can be used for $\mathbf{\Sigma}$, this study focuses on the use of Jeffreys prior and the inverse Wishart prior.

Jeffreys prior Jeffreys prior (e.g., Gelman et al., 2013; Jeffreys, 1946) is a type of noninformative prior, for it does not incorporate extra information other than that from the data to the posterior distribution. For a model with a vector of parameters $\boldsymbol{\zeta}$, its Jeffreys prior is defined through

$$\pi_J(\boldsymbol{\zeta}) \propto \sqrt{\det(\mathcal{I}(\boldsymbol{\zeta}))} \tag{7}$$

where $\mathcal{I}(\boldsymbol{\zeta})$ is the Fisher-information matrix. A Wishart likelihood is thus

$$\pi_J(\varSigma) = |\varSigma|^{-(p+1)/2} \tag{8}$$

where p is the number of variables. This prior distribution was firstly developed by Jeffreys (1961) for $p = 1, 2$, and later was generalized to arbitrary p (e.g., Geisser, 1965; Geisser & Cornfield, 1963; Villegas, 1969).

With the Jeffreys prior, the posterior distribution of $\varSigma$ is

$$\pi(\mathbf{\Sigma}|\mathbf{S}) \propto |\varSigma|^{-(n+p+1)/2} \exp[-\frac{1}{2}\text{tr}(n\mathbf{S}\varSigma^{-1})], \tag{9}$$

which is an Inverse Wishart (IW) distribution with degrees of freedom n and scale matrix $n\mathbf{S}$, denoted as IW(n,$n\mathbf{S}$) in this study.

Inverse Wishart prior The Inverse Wishart prior is a conjugate prior and is widely used in practice. The Inverse Wishart prior IW(m,$\mathbf{V}$) has the following probability density function,

$$P(\varSigma|\mathbf{V}, m) = \frac{|\mathbf{V}|^{m/2}}{2^{mp/2}\varGamma_p(\frac{m}{2})}|\varSigma|^{-\frac{m+p+1}{2}} \exp[-\frac{1}{2}\text{tr}(\mathbf{V}\varSigma^{-1})]. \tag{10}$$

Note that with Jeffreys prior, the posterior distribution of the population covariance matrix is an Inverse Wishart distribution with the degrees of freedom being the sample size n and the scale matrix being the sums of squares $n\mathbf{S}$. Comparing the form in Eqn (10) to that in Eqn (9), we notice the use of an Inverse Wishart prior IW(m, $\mathbf{V}$) in an analysis is theoretically equivalent to provide m additional observations with sums of squares $\mathbf{V}$ to the estimation of Σ.

With the Inverse Wishart prior IW(m,$\mathbf{V}$), the posterior distribution is also an Inverse Wishart distribution with the following kernel,

$$\begin{aligned} P(\Sigma|\mathbf{S}) &\propto |\Sigma|^{-n/2} \exp[\frac{1}{2}\mathrm{tr}(n\mathbf{S}\Sigma^{-1})]|\Sigma|^{-\frac{m+p+1}{2}} \exp[-\frac{1}{2}\mathrm{tr}(\mathbf{V}\Sigma^{-1})] \\ &= |\Sigma|^{-\frac{n+m+p+1}{2}} \exp(-\frac{1}{2}\mathrm{tr}[(n\mathbf{S}+\mathbf{V})\Sigma^{-1})]. \end{aligned} \tag{11}$$

Thus it is the Inverse Wishart distribution IW$(n+m, n\mathbf{S}+\mathbf{V})$ with the degrees of freedom $n+m$ and the scale matrix $n\mathbf{S}+\mathbf{V}$. Note that if we set $\mathbf{V} = m\mathbf{S}$, then the prior represents $\frac{100m}{n}\%$ of information of the data. Therefore, the amount of information in the prior can be easily quantified.

The posterior mean and mode are $\frac{n\mathbf{S}+\mathbf{V}}{n+m-p-1}$ and $\frac{n\mathbf{S}+\mathbf{V}}{n+m+p+1}$, respectively. For a fixed m, both the posterior mean and mode as well as the sample covariance matrix $\mathbf{S}$ will converge to the population covariance matrix Σ asymptotically. Thus, when the sample size is large, all three estimates $\mathbf{S}$, $\frac{n\mathbf{S}+\mathbf{V}}{n+m-p-1}$, and $\frac{n\mathbf{S}+\mathbf{V}}{n+m+p+1}$ are similar. For a given n, a larger m indicates that the prior influences the posterior distribution more.

In the existing Bayesian SEMs using an Inverse Wishart prior, the scale matrix $\mathbf{V}$ is often chosen to be an identical matrix $\mathbf{I}$. However, we could change the Inverse Wishart prior to an informative prior by using a specific scale matrix $\mathbf{V}$. For instance, if we know the estimated sums of squares from another study, we can use it as our scale matrix in the Inverse Wishart prior. In addition, if one does not want to use prior information, one can set $m = 0$ and $\mathbf{V} = \mathbf{0}$. Note that the density function is not proper any more but the prior becomes the Jeffreys prior. In this sense, the Jefferys prior can be regarded as a special case of Inverse Wishart prior. Hence, in the rest of the paper, we only focus on the Inverse Wishart prior. Thus, the posterior distribution of the population covariance parameter is an inverse-Wishart distribution IW$(n+m, n\mathbf{S}+\mathbf{V})$.

3.3 Posterior distribution of $\boldsymbol{\theta}$

With the posterior distribution of Σ, i.e., IW$(n+m, n\mathbf{S}+\mathbf{V})$, we can further get a posterior distribution of model parameters. Let g be a function that maps a Σ to a $\boldsymbol{\theta}_\Sigma$. Among the so many possible candidates, we define $g(\boldsymbol{\Sigma})$ as the $\boldsymbol{\theta}$ that minimizes the distance between $\Sigma(\boldsymbol{\theta})$ and Σ,

$$\boldsymbol{\theta}_\Sigma = g(\Sigma) = \mathrm{argmin}(\log|\Sigma(\boldsymbol{\theta})| + \mathrm{tr}(\Sigma\Sigma^{-1}(\boldsymbol{\theta})) - \log|\Sigma|) \tag{12}$$

This step is analogous to the maximum likelihood (ML) estimation of SEM models, in which the sample covariance matrix $\mathbf{S}$ is used as the estimate of the population covariance matrix Σ. However, in Bayesian SEM, the population covariance matrix Σ is a random variable and can take values from the space Ω_{Σ}. We therefore obtain a $\boldsymbol{\theta}_{\Sigma}$ for a covariance matrix from the space Ω_{Σ}.

By aplplying the mapping function $g(\cdot)$ to the posterior distribution $P(\Sigma|\mathbf{S})$, we could get a posterior distribution of $\boldsymbol{\theta}$ called $P(\boldsymbol{\theta}|\mathbf{S})$. Although we do not know the exact form of $P(\boldsymbol{\theta}|\mathbf{S})$,we could draw samples from it. To do so, we first drawn B independent samples of Σ denoted by $\Sigma_1, \Sigma_2, \cdots, \Sigma_B$ from $P(\Sigma|\mathbf{S})$. We then map each of such Σ_bs to $\boldsymbol{\theta}_b$s using the function g defined in (12).

$\boldsymbol{\theta}$-mean estimate With B samples $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \cdots, \boldsymbol{\theta}_B$ obtained using the procedure described above, an estimate of posterior mean, called $\boldsymbol{\theta}$-mean, is defined as the average of samples of $\boldsymbol{\theta}$,

$$\hat{\boldsymbol{\theta}}_{mean} = \frac{1}{B}\sum_{b=1}^{B}\boldsymbol{\theta}_b. \tag{13}$$

In addition to the $\boldsymbol{\theta}-$mean estimate, there are some other parameter estimates of practical interest. Depending on how the estimates of the population covariance matrix is constructed from its posterior distribution, we can have different forms of estimates as discussed below.

$\boldsymbol{\Sigma}$-mode estimate and $\boldsymbol{\Sigma}$-mean estimate In Bayesian inference, the maximum a posteriori (MAP) estimate is of great interest, because it is the mode of a posterior distribution. With an Inverse Wishart prior IW$(m, \mathbf{V})$, the posterior mode of Σ is $\Sigma_{mode}=\frac{nS+\mathbf{V}}{m+n+p+1}$. We then find its corresponding $\boldsymbol{\theta}$ using Equation (12). The resulted $\boldsymbol{\theta}$ is an estimate for the parameter $\boldsymbol{\theta}$ and is called Σ-mode estimate in this study.

Similarly, we can also get the posterior mean of Σ, $\Sigma_{mean} = \frac{ns+\mathbf{V}}{m+n-p-1}$ and then finds out its corresponding $\boldsymbol{\theta}$.We will call it the Σ-mean estimate in the rest of this study. It is interesting to note that when the IW$(\mathbf{V} = \mathbf{0}, m = p+1)$, the Σ_{mean} is the same as the sample covariance matrix and $\Sigma-$mean estimate will concide the ML estimates of the model parameters.

Highest posterior density (HPD) credible intervals Besides the point estimates, the posterior credible intervals can also be formed using the posterior samples $\boldsymbol{\theta}_b, b = 1, \ldots B$. For each element θ in the vector of parameters $\boldsymbol{\theta}$ and a desired level α between 0 and 1, the $100\alpha\%$ HPD credible interval is denoted by $[L_\alpha, U_\alpha]$ as the one with smallest width among all the intervals containing α proportion of samples.

So far, we have explained how to specify priors on the population covariance parameter, how to obtain posterior samples for model parameters $\boldsymbol{\theta}$. To obtain posterior statistics, we introduced three point estimates and the HPD credible intervals. Since the proposed approach specifies prior on the population covariance matrix, we call it "Covariance matrix prior Bayesian SEM (CP-BSEM)" in contrast to the traditional Bayesian SEM approaches (T-BSEM)

4 Comparison of CP-BSEM and T-BSEM

The CP-BSEM approach has several distinct features from the T-BSEM method. The first one is the prior specification. While using the CP-BSEM approach, we specify a prior on the population covariance matrix parameter Σ. The posterior distributions of model parameters are transformed from the posterior distribution of the population covariance matrix $P(\Sigma|data)$. We can notice that the CP-BSEM approach reduces the burden of specifying priors and posterior diagnostics.

However, an essential question that arises is whether the CP-BSEM approach can lead to comparable posterior inference on $\boldsymbol{\theta}$ as to a T-BSEM approach. While using a T-BSEM approach, an implict assumption is that the model is a correct model and there is no model misfit. In the following, we will show that the CP-BSEM is coupling to a T-BSEM approach with a certain prior specification when the model is a true model.

Let $P_{\text{IW}}(\Sigma)$ be the density function of the Inverse Wishart prior $\text{IW}(m, \mathbf{V})$ for the population covariance matrix parameter Σ and $g(\cdot)$ be a function that maps a Σ to a $\boldsymbol{\theta}$ as defined by Equation (12). When applying the g function to the entire space of Σ, i.e., Ω_Σ,with the probability density function $P_{\text{IW}}(\Sigma)$, we could get a probability distribution function on $\boldsymbol{\theta}$. Specifically,

$$g : \big(\Omega_\Sigma, P_{\text{IW}}\big) \rightarrow \big(\Omega_{\boldsymbol{\theta}}, P_{\text{IW}}(\Sigma(\boldsymbol{\theta}))\big)$$

where $P_{\text{IW}}(\Sigma(\boldsymbol{\theta}))$ is the probability distribution on model parameters $\boldsymbol{\theta}$ that is transformed from P_{IW} by the mappling function $g(\cdot)$.

Let $P(\Sigma|data)$ be the posterior distribution and it is still an Inverse Wishart distribution $\text{IW}(m + n, n\mathbf{S} + \mathbf{V})$. We can transform it to a distribution on $\boldsymbol{\theta}$, named as $P_\Sigma(\boldsymbol{\theta}|data)$, using the mapping function $g(\cdot)$,

$$\begin{aligned} P_\Sigma(\boldsymbol{\theta}|data) &= P(g^{-1}(\boldsymbol{\theta})|data) \\ &= P(\Sigma|data) \\ &= P(data|g(\Sigma))P_{\text{IW}}(\Sigma) \\ &= P(data|\Sigma(\boldsymbol{\theta}))P_{\text{IW}}(\Sigma(\boldsymbol{\theta})) \end{aligned} \tag{14}$$

where $P(data|\Sigma(\boldsymbol{\theta}))$ is the likehood for given $\boldsymbol{\theta}$ and $P_{\text{IW}}(\Sigma(\boldsymbol{\theta}))$ is a prior distribution on model parameter $\boldsymbol{\theta}$. Therefore, the distribution on $\boldsymbol{\theta}$ transformed from the posterior distribution of Σ is actually a posterior distribution of $\boldsymbol{\theta}$ with the given prior distribution $P_{\text{IW}}(\Sigma(\boldsymbol{\theta}))$ using the T-BSEM approach.

As discussed above, the proposed CP-BSEM approach can be coupled to a T-BSEM, therefore it can achieve comparable inference with the T-BSEM approach. Moreover, it eases up the burden of prior specification and has better interpretation of the prior information, and it does not need posterior diagnosis.

5 Empirical Study

To show how to use the newly proposed CP-BSEM approach, we analyzed the data from Holzinger and Swineford (1939), which includes measures on 19 tests from 145 eighth grade students in the Grant-White School and 156 students from the Pasteur School. As an illustration, we focus on the analysis of the data from the Grant-White School. The 19 tests were intended to measure four attributes: spatial ability, verbal ability, process speed, and working memory. Therefore, we fitted a four-factor confirmatory factor model shown in Figure 1. In identifying the model, we fixed one factor loading for each factor to be 1 and freely estimated the factor variances and covariances as shown in the path diagram.

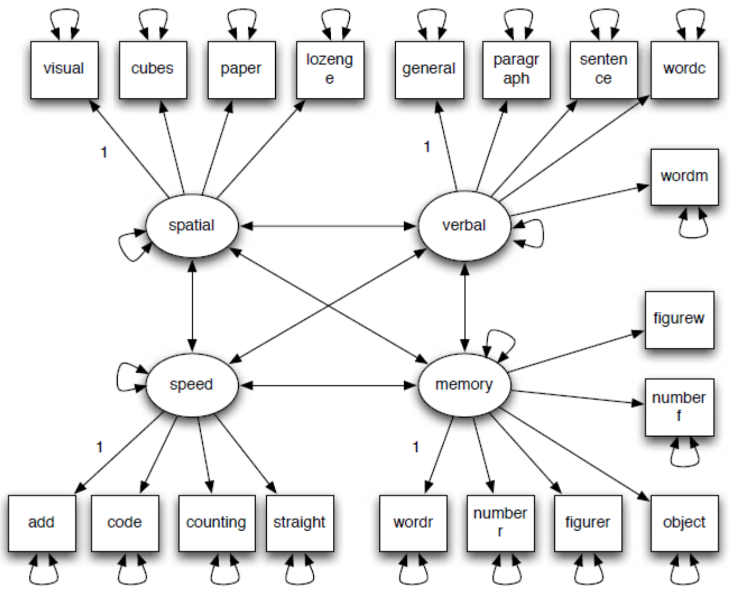


Figure 1. Path diagram of the CFA model fitted to the Holzinger data. The factor loading of each factor to its first indicator is fixed to be 1 and the factor variances are freely estimated.

For the purpose of illustration, we analyzed the data using two different priors. The first one is a noninformative prior $\mathrm{IW}(19, \mathbf{I})$. The second prior was an informative prior formed based on the data from the Pasteur school. With the noninformative prior, the posterior distribution for the population covariance matrix is $\mathrm{IW}(145 + 19, 145\mathbf{S}_{GW} + \mathbf{I})$, with $\mathbf{S}_{GW}$ being the sample covariance matrix of the data from Grant-White School.

The informative prior distribution was specified as the posterior distribution obtained based on the data from the Pasteur School with 156 participants. In the analysis of the Pasterur School data, suppose the same noninformative prior $\mathrm{IW}(19, \mathbf{I})$ was used, which led to the posterior distribution $\mathrm{IW}(156{+}19{,}156\mathbf{S}_P + \mathbf{I})$, with $\mathbf{S}_P$ being the sample covariance matrix of the data from Pasteur School.

Using it as the prior distribution for the Grant-White school data analysis, we redid the analysis of the Grant-White School data and the resulting posterior distribution is $\text{IW}(145 + 156 + 19, 145\mathbf{S}_{GW} + 156\mathbf{S}_P + \mathbf{I})$. Note that in practical data analysis, the raw data to form the prior are often not available. Therefore, the current way for incorporating prior information is even more practical to substantive researchers.

The model parameter estimates using the two priors based on Σ-mean together with the 95% HPD credible intervals and the width of the intervals are summarized in Table 1. Note that the unique factor variances are not reported to save space. Clearly, all the 95% HPD intervals excluded 0 and, therefore, one may conclude all the parameters were statistically significant from 0 in the frequentist hypothesis testing sense. The parameter estimates obtained using noninformative and informative priors were quite different. In addition, the HPD intervals using the informative prior were narrower consistently than those from using the noninformative prior. According to our simulation, when the $\text{IW}(19, \mathbf{I})$ was used, the estimates should be close to MLE. Therefore, the difference was because of the use of prior information in the Bayesian estimation process. The choice between the two sets of estimates depends on whether we wanted to use the prior information or not. If the inference was supposed to be based on only the data from the Grant-White School, the noninformative prior should be used. If the inference was designed to also combine the information from the Pasteur School, the informative prior should be adopted. This empirical example illustrated that it is possible either to use or not to use prior information.

6 Simulation Study

The purpose of the simulation study are twofold. First, we would like to evaluate the performance of the CP-BSEM approach. We will compare the three types of point estimates in terms of how well they could recover the true parameter values. We will also evaluate the HPD credible intervals to see whether it has good coverage rates. Second, we will compare the CP-BSEM approach with two other competing approaches: The ML approach and the T-BSEM approach. Therefore, we will also report the results of the ML method and the C-BSEM approach with default settings, which are implemented in R packages *lavaan* (Rosseel, Oberski, Byrnes, Vanbrabant, & Savalei, 2013) and *blavaan* (Merkle & Rosseel, 2015).

6.1 Simulation design

The simulation study is designed based on the 4-factor confirmatory factor model with 19 normally distributed indicators as used in the empirical data analysis presented later. Let Z be the normally distributed random vectors of 19 variables with mean $\mathbf{0}$ and the four latent factors be $\mathbf{f} = (f_1, f_2, f_3, f_4)'$. The factor model is

$$Z = \mathbf{\Lambda f} + \boldsymbol{\varepsilon} \tag{15}$$

Table 1. Model parameter estimates of the confirmatory factor model using the Holzinger data

	Noninformative Prior						Informative Prior				
	ML	$\boldsymbol{\theta}_{\boldsymbol{\Sigma}_{mode}}$	$\boldsymbol{\theta}_{\boldsymbol{\Sigma}_{mean}}$	$\boldsymbol{\theta}_{mean}$	95% CI	W	$\boldsymbol{\theta}_{\boldsymbol{\Sigma}_{mode}}$	$\boldsymbol{\theta}_{\boldsymbol{\Sigma}_{mean}}$	$\boldsymbol{\theta}_{mean}$	95% CI	W
					Fator Loading						
visual	0.689	0.612	0.691	0.688	0.513 0.866	0.352	0.708	0.757	0.755	0.62 0.889	0.269
cubes	0.477	0.423	0.479	0.476	0.292 0.659	0.368	0.435	0.465	0.466	0.322 0.605	0.283
paper	0.541	0.48	0.543	0.541	0.365 0.719	0.354	0.474	0.506	0.504	0.373 0.636	0.264
lozenge	0.674	0.598	0.676	0.675	0.507 0.856	0.348	0.623	0.665	0.666	0.528 0.804	0.277
general	0.805	0.715	0.808	0.806	0.674 0.957	0.283	0.793	0.847	0.846	0.748 0.954	0.206
paragrap	0.817	0.725	0.819	0.817	0.679 0.961	0.282	0.784	0.838	0.838	0.736 0.942	0.207
sentence	0.834	0.74	0.837	0.834	0.702 0.978	0.276	0.824	0.881	0.881	0.782 0.988	0.207
wordc	0.693	0.615	0.696	0.695	0.551 0.849	0.298	0.686	0.733	0.733	0.63 0.844	0.214
wordm	0.840	0.745	0.843	0.842	0.707 0.987	0.281	0.812	0.868	0.868	0.767 0.974	0.207
add	0.649	0.576	0.651	0.65	0.453 0.844	0.39	0.588	0.628	0.626	0.484 0.766	0.282
code	0.694	0.616	0.696	0.691	0.52 0.858	0.338	0.708	0.757	0.756	0.629 0.888	0.259
counting	0.697	0.619	0.699	0.699	0.524 0.877	0.353	0.596	0.637	0.637	0.496 0.778	0.282
straight	0.745	0.662	0.748	0.747	0.566 0.93	0.364	0.63	0.673	0.674	0.537 0.812	0.275
wordr	0.515	0.457	0.516	0.515	0.314 0.707	0.394	0.549	0.587	0.585	0.446 0.722	0.276
numberr	0.516	0.458	0.518	0.517	0.342 0.698	0.356	0.497	0.531	0.53	0.395 0.669	0.274
figurer	0.595	0.528	0.597	0.597	0.415 0.78	0.365	0.58	0.619	0.617	0.477 0.764	0.286
object	0.627	0.556	0.629	0.626	0.443 0.81	0.367	0.578	0.618	0.616	0.476 0.754	0.279
numberf	0.648	0.575	0.65	0.647	0.463 0.83	0.367	0.538	0.575	0.574	0.44 0.714	0.274
figurew	0.482	0.428	0.484	0.481	0.303 0.666	0.363	0.461	0.493	0.492	0.352 0.629	0.277
					Factor Covariance						
spatial ∼verbal	0.594	0.594	0.594	0.593	0.425 0.732	0.307	0.504	0.504	0.502	0.369 0.616	0.248
spatial ∼ speed	0.583	0.583	0.582	0.577	0.336 0.771	0.435	0.469	0.469	0.467	0.302 0.619	0.316
spatial ∼ memory	0.635	0.635	0.635	0.632	0.409 0.804	0.395	0.499	0.499	0.495	0.336 0.644	0.308
verbal ∼ speed	0.467	0.467	0.467	0.463	0.276 0.615	0.339	0.464	0.464	0.462	0.339 0.576	0.237
verbal ∼ memory	0.496	0.497	0.497	0.496	0.319 0.647	0.328	0.347	0.347	0.346	0.206 0.474	0.268
speed ∼ memory	0.605	0.605	0.605	0.599	0.405 0.76	0.355	0.552	0.552	0.55	0.408 0.675	0.267

Note. The notation $\lambda_{j,k}$ represents the factor loading from the jth factor to the kth indicator; $\sigma^2_{f,j,l}$ is the factor covariance of the jth factor and the lth factor.

where $\mathbf{\Lambda}$ is a 19 by 4 factor loading matrix, and $\boldsymbol{\varepsilon} = (\varepsilon_1, \varepsilon_2, \cdots, \varepsilon_{19})'$ is the unique factor score. The factor loading matrix has the following form,

$$\mathbf{\Lambda} = \begin{pmatrix} \boldsymbol{\lambda_1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\lambda_2} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \boldsymbol{\lambda_3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \boldsymbol{\lambda_4} \end{pmatrix}$$

with $\boldsymbol{\lambda_1} = (1, .693, .785, .978)'$, $\boldsymbol{\lambda}_2 = (1, 1.015, 1.036, 0.861, 1.043)'$, $\boldsymbol{\lambda_3} = (1, 1.069, 1.075, 1.149)'$, $\boldsymbol{\lambda}_4 = (1, 1.004, 1.156, 1.218, 1.259, 0.936)'$. The covariance matrix of the latent factors is

$$\mathbf{\Phi} = \begin{pmatrix} .473 & .328 & .260 & .224 \\ .328 & .646 & .243 & .205 \\ .260 & .243 & .420 & .201 \\ .224 & .205 & .201 & .264 \end{pmatrix},$$

and the uniqueness factor covariance matrix $\mathbf{\Psi}$ is a diagonal matrix with diagonal elements (0.517, 0.763, 0.699, 0.537, 0.344, 0.325, 0.297, 0.511, 0.287, 0.570, 0.511, 0.505, 0.436, 0.726, 0.724, 0.637, 0.598, 0.572, 0.759). All parameter values are chosen to reflect the parameter estimates in the empirical data analysis.

Based on the population model, we generate 1000 data sets for each of the following sample sizes: 100, 150, 250, 300 and 500. Then, we fit the model to the generated data using ML, two-stage Bayesian, and traditional Bayesian methods. In the two-stage Bayesian method, the “noninformative” prior IW(19, $\mathbf{I}$) is used since we want to evaluate the parameter bias and estimation efficiency. In addition, 10000 covariance matrices are sampled independently from the posterior distribution of Σ to form the credible intervals. For traditional Bayesian estimates, the R package *blavaan* is used and the following default priors are adopted in the estimation:

$$\begin{aligned} \text{factor loadings } \lambda_{j,k} &\overset{iid}{\sim} \mathrm{N}(0, 10^4) \\ \text{factor covariance matrix } \mathit{\Phi} &\sim \mathrm{IW}(5, \mathbf{I}) \\ \text{unique factor variances } \sigma_k^2 &\overset{iid}{\sim} \mathrm{IG}(1, 0.5) \end{aligned}$$

where $\lambda_{j,k}$ represents the factor loading from the jth factor to the kth indicator; σ_k^2 is the unique factor variance of the kth indicator. Finally, MLE is obtained using the R package *lavaan*.

6.2 Evaluation criteria

To evaluate the performance of each method, we report the relative bias, coverage rates of the HPD credible intervals/confidence intervals, and mean squared errors for the model parameters.

Relative bias Let θ represent a parameter or its true value. The relative bias is defined as the percent ratio of the discrepancy between the estimate and the true value with respect to the true value of a parameter:

$$\text{relative bias}_\theta = \begin{cases} \frac{\bar{\theta}-\theta}{|\theta|} \times 100\% & \text{if } \theta \neq 0 \\ (\bar{\theta}-\theta) \times 100\% & \text{otherwise} \end{cases}, \tag{16}$$

where $\bar{\theta}$ is the average of the estimates in R, which is 1000 successful replications in our simulation,

$$\bar{\theta} = \frac{1}{R}\sum_{r=1}^{R}\hat{\theta}_r.$$

with $\hat{\theta}_r$ denoting the parameter estimate in the rth replication.

Mean squared error (MSE) The mean squared errors (MSE) are calculated as,

$$\begin{aligned} \text{MSE}_\theta &= \frac{1}{R}\sum_{r=1}^{R}(\hat{\theta}_r-\theta)^2 \\ &= \frac{1}{R}\sum_{r=1}^{R}(\hat{\theta}_r-\bar{\theta})^2 + \frac{1}{R}\sum_{r=1}^{R}(\bar{\theta}-\theta)^2 \end{aligned} \tag{17}$$

which is the sum of the variance and squared biases of the parameter estimates.

Coverage rate (CR) The coverage rate of the 95% HPD credible interval for Bayesian and confidence interval for MLE represents the proportion of the intervals covering the true parameter value. Mathematically, if $[L^r_{0.95}, U^r_{0.95}]$ is the interval in the rth replication, the coverage rate (CR) is calculated as

$$\text{CR}_\theta = \frac{1}{R}\sum_{r=1}^{R} I(\theta \in [L^r_{0.95}, U^r_{0.95}]).$$

where $I(\cdot)$ is the index function with value 1 if the interval covers the true value and 0, otherwise. A CR around 0.95 implies that the defined 95% interval performs well.

6.3 Simulation results

We now present the results on relative biases, coverage rates, and mean squared errors from our simulation. There are 44 parameters grouped into three types – 15 factor loadings, 19 unique factor variances, and 10 factor covariances. Since we found that the influence of estimation methods on each type of parameters is similar, we only report the average relative biases, coverage rates, and mean squared errors for the three types of parameters to save space. For relative bias, we calculate the average based on the absolute values because bias can be positive or negative.

Average absolute relative bias The average absolute relative biases for factor loadings, factor covariance matrix and unique factor variances are provided in Table 2. For the two-stage Bayesian, the three types of parameter estimates are presented. For the traditional Bayesian, the posterior mean and median are obtained using *blavaan*.

Overall, the bias decreased as the sample size increased for parameter estimates. For the factor loadings, the bias for Σ-mean estimates was small even when the sample size was 100 and was almost the same as the ML estimates. $\boldsymbol{\theta}$-mean had larger bias than Σ-mean estimates but overall was better than the traditional method. Although $\mathbf{\Sigma}$-mode estimates had small bias for factor loadings but had large bias for factor covariance and unique factor variances. This indicates that in using the two-stage Bayesian for parameter estimates, either Σ-mean or $\boldsymbol{\theta}$-mean estimates should be preferred.

Comparing the two-stage method with the traditional Bayesian method, it was clear that the two-stage method provided less biased parameter estimates using the chosen prior. This is because we had better control of the prior information in the two-stage method.

Table 2. Absolute relative biases for factor loadings, factor covariances, and unique factor variances

		CP-BSEM			T-BSEM	
N	ML	Σ-mode	Σ-mean	θ-mean	Mean	Median
			Factor loading			
100	2.421	2.419	2.419	*8.300*	*8.691*	*6.87*
150	1.357	1.357	1.357	3.329	*6.246*	4.969
200	0.926	0.926	0.926	2.306	4.941	3.949
250	0.958	0.958	0.958	1.992	4.295	3.483
300	0.748	0.748	0.748	1.579	3.626	2.940
500	0.461	0.461	0.461	0.924	2.354	1.933
			Factor covariance			
100	1.263	**28.694**	0.733	1.853	*9.706*	**12.613**
150	1.101	**20.181**	1.366	1.885	*6.37*	*8.393*
200	0.99	**16.882**	0.690	1.142	*6.228*	*7.752*
250	0.698	**13.764**	0.598	0.891	4.969	*6.199*
300	0.441	**11.575**	0.435	0.712	4.086	*5.118*
500	0.342	*7.115*	0.403	0.535	2.502	3.124
			Unique factor variance			
100	2.352	**28.294**	0.915	1.069	2.292	1.242
150	1.573	**20.813**	0.607	0.858	1.535	0.968
200	1.165	**16.447**	0.542	0.621	1.199	0.774
250	0.993	**13.655**	0.329	0.439	0.921	0.57
300	0.703	**11.53**	0.378	0.321	0.896	0.493
500	0.49	*7.315*	0.228	0.258	0.503	0.364

Note. A bold numbers means an average absolute relative bias larger than 10% and an Italic number represents an average absolute relative bias larger than 5% but smaller than 10%.

Mean squared error The mean squared errors for the parameters are provided in Table 3. Similar to the bias, the MSE also decreased as the sample size increased for all parameters regardless of the estimation methods. The mean squared errors were mostly comparable with two notable observations. First, the MSE for MLE and Σ-mean were almost identical. This again suggested the control of prior information. Second, the MSE from the traditional Bayesian method was smaller than MLE. This is because the use of the prior information. Therefore, in terms of both bias and MSE, the two-stage method has better control of the influence of the prior information.

Table 3. Mean squared errors for factor loadings, factor covariances, and unique factor variances

MSE×100		CP-BSEM			T-BSEM	
N	ML	Σ-mode	Σ-mean	θ-mean	Mean	Median
			Factor loading			
100	5.742	5.737	5.737	20.775	5.271	4.765
150	3.555	3.553	3.553	4.377	3.415	3.147
200	2.466	2.466	2.466	2.792	2.517	2.343
250	2.027	2.027	2.027	2.220	2.078	1.951
300	1.636	1.635	1.635	1.763	1.687	1.592
500	0.941	0.941	0.941	0.979	0.984	0.942
			Factor covariance			
100	0.953	1.517	0.972	0.970	0.816	0.859
150	0.672	0.940	0.679	0.682	0.593	0.613
200	0.497	0.698	0.500	0.501	0.463	0.477
250	0.388	0.528	0.391	0.392	0.362	0.372
300	0.332	0.426	0.335	0.336	0.313	0.319
500	0.197	0.236	0.198	0.198	0.190	0.194
			Unique factor variance			
100	0.935	3.061	0.935	0.910	0.924	0.891
150	0.608	1.783	0.608	0.602	0.605	0.590
200	0.459	1.196	0.459	0.453	0.456	0.449
250	0.367	0.874	0.366	0.364	0.365	0.360
300	0.306	0.667	0.307	0.304	0.306	0.303
500	0.179	0.328	0.180	0.179	0.179	0.179

Note .The reported numbers are the average MSEs multiplied by 100.

Coverage rate The coverage rates for model parameters are displayed in Table 4. Overall, the coverage rates were close to the nominal level 0.95 except for the factor covariances when the traditional Bayesian method was used.

In summary, our two-stage Bayesian method can obtain results similar to ML method by controlling the prior information. Comparing to the T-BSEM method, it also offers better control of prior information.

Table 4. Average coverage rates for factor loadings, factor covariance parameters, and unique factor variances

	N	ML	CP-BSEM	T-BSEM
Factor loading	100	94.29	95.21	94.43
	150	94.43	95.03	94.48
	200	94.85	95.03	94.54
	250	94.61	94.98	94.60
	300	94.86	94.90	94.57
	500	95.41	95.18	95.01
Factor covariance	100	92.70	94.71	**90.34**
	150	93.36	94.47	**91.27**
	200	93.41	94.33	**91.30**
	250	94.35	95.21	92.68
	300	93.82	94.39	**92.28**
	500	94.73	94.93	93.61
Unique factor variance	100	**92.22**	94.53	95.30
	150	93.31	94.72	95.35
	200	93.65	94.70	95.35
	250	93.87	94.52	94.97
	300	94.10	94.67	95.08
	500	94.46	94.86	95.18

Note. A bold number represents an average coverage rate smaller than 92.5%.

7 Discussion and Conclusion

In traditional Bayesian SEM, a prior needs to be specified for each individual or individual set of model parameters. Due to the complexity of SEM models and the diverse features of different types of model parameters, specifying priors is not an easy task, especially if one would like to control or utilize prior information. To get parameter estimates, MCMC procedures are often used and the convergence diagnostics of MCMC samples are always required, which is usually hard for researchers conducting applied researches.

In the present study, an alternative Bayesian procedure, i.e., CP-BSEM, is proposed to assist researchers conducting Bayesian statistical inference of SEMs. It has several distinct benefits over the traditional Bayesian procedures. First, the prior information is only required for the population covariance matrix parameter Σ. Using the Inverse Wishart prior, we can control the prior information ranging from noninformative to very informative. The information can be conveniently controlled by varying its degrees of freedom and scale matrix. For instance, when both the degrees of freedom and scale matrix are set at 0, it becomes the Jeffreys prior for covariance matrix analysis with normal data. Increasing the degrees of freedom and/or using a special scale matrix, we could make the Inverse Wishart prior informative. The amount of information can also be directly compared to the data at hand.

Unlike the traditional Bayesian SEM, the CP-BSEM procedure does not require convergence diagnostics. With the use of the Inverse Wishart prior, the posterior of the population covariance matrix still follows an Inverse Wishart distribution. Therefore, independently and identically distributed samples of covariance matrix can be drawn from the posterior distribution directly. The corresponding samples of parameter estimates are thus independently and identically distributed, too. Since the samples are identically distributed, convergence diagnostics are not needed any more. The independence among the samples enables us to use relatively fewer samples to get reliable inferences than the traditional Bayesian SEM.

Results from our simulation study show that the CP-BSEM procedure works well in estimating structural equation models in general. Among the three point estimates, the Σ-mean estimates, obtained by fitting the SEM model to the posterior mean of the covariance matrix, is recommended. They have ignoble relative biases ($< 5\%$) with results close to MLE. In addition, the credible interval has good coverage rates. We also notice that the traditional Bayesian SEM had slightly smaller MSEs than our two-stage procedure and MLE. This is due to more prior information involved in their prior distributions. By changing the prior distribution, traditional Bayesian SEM can also reduce the influence of prior distributions. However, as we have pointed out, controlling the prior information in traditional Bayesian analysis can be difficult, especially for applied researchers.

Compared to the traditional Bayesian method, the performance of the CP-BSEM procedure is less affected by the model complexity. Because the prior information is put on the population covariance matrix, it is independent to the model structure. Therefore, we could extend our results to a more general SEM model.

The CP-BSEM procedure is flexible to control prior information. In our empirical example, we demonstrated the use of informative prior, which was the posterior distribution of the covariance matrix from another study, also an Inverse Wishart distribution. Hence, the CP-BSEM approach can be used to conduct meta-analysis by combining several related studies in the SEM framework. Instead of combing model parameter estimates of every single study, one could combine the covariance matrices of different studies, in which the posterior distribution of the previous study will work as the prior distribution in the new study. One immediate benefit is that the inference will focus on the overall covariance matrix, but not the individual model parameters in each study. As a result, it has special advantages in combining studies without a common model structure.

The CP-BSEM approach is closely related to the parametric bootstrap technique for SEM. Covariance matrices are drawn from its posterior distribution repeatedly, and samples of parameter estimates are obtained by minimizing the descrepancy between the model implied covariance matrix to the sampled covariance matrices. The HPD credible intervals formed based on the samples of model parameters are, therefore, have the similar meaning to the bootstrap confidence intervals. However, in bootstrap, no prior information is allowed but the CP-BSEM approach can easily utilize prior information. Therefore, it could be particularly useful, when the original data set has a small sample size and the actual model parameters estimates are hard to obtain.

Even with the advantages, we want to note that the CP-BSEM approach can still not replace the traditional Bayesian SEM. For example, currently, missing data and non-normal data cannot be handled yet. Therefore, to increase the impact of the CP-BSEM and to help the adoption of the Bayesian methods, the CP-BSEM approach can be expanded in the following aspects. First, the present version of CP-BSEM procedure focuses on SEMs without mean structures. Extending the method to include the mean structure can make it possible to conduct growth curve analysis and multiple group analysis. Second, how to handle missing data in the CP-BSEM procedure should be investigated. Third, ways should be evaluated to handle non-normal data in the CP-BSEM procedure. Fourth, in traditional Bayesian SEM, deviance information criterion and posterior predictive p-values have been used for model fit evaluation. They can also be incorporated in the CP-BSEM approach.

Acknowledgment

The study was supported by a grant from the Institute of Education Sciences (R305D210023).

References

Anderson, J., & Gerbing, D. W. (1988). Structural equation modeling in practice: A review and recommended two-step approach. *Psychological Bulletin*, *103*(3), 411–423. doi: https://doi.org/10.1037/0033-2909.103.3.411

Bentler, P., & Dudgeon, P. (1996). Covariance structure analysis: Statistical practice, theory, and direction. *Annual l Review of Psychology*, *47*(1), 563 –592. doi: https://doi.org/10.1146/annurev.psych.47.1.563

Bentler, P. M., & Yuan, K.-H. (1999). Structural equation modeling with small samples: Test statistics. *Multivariate Behavioral Research*, *34*(2), 181–197. doi: https://doi.org/10.1207/S15327906Mb340203

Boker, S. M., & McArdle, J. J. (2005). Path analysis and path diagrams. *Wiley StatsRef: Statistics Reference Online*, *3*, 1529 –1531. doi: https://doi.org/10.1002/9781118445112.stat06517

Bollen, K. (1989). *Structure equations with latent variables.* New York: Wiley.

Brooks, S. P., & Roberts, G. O. (1998). Convergence assessment techniques for markov chain monte carlo. *Statistics and Computing*, *8*(4), 319–335. doi: https://doi.org/10.1023/A:1008820505350

Depaoli, S., Liu, H., & Marvin, L. (2021). Parameter specification in bayesian cfa: An exploration of multivariate and separation strategy priors. *Structural Equation Modeling: A Multidisciplinary Journal*, *28*(5), 699–715. doi: https://doi.org/10.1080/10705511.2021.1894154

Geisser, S. (1965). Bayesian estimation in multivariate analysis. *The Annals of Mathematical Statistics*, *36*(1), 150–159.

Geisser, S., & Cornfield, J. (1963). Posterior distributions for multivariate normal parameters. *Journal of the Royal Statistical Society. Series B (Methodological)*, 368–376. doi: https://doi.org/10.1080/01621459.1990.10476213

Gelfand, A., & Smith, A. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, *85*, 398–409. doi: https://doi.org/10.1080/01621459.1990.10476213

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). (3rd ed.). CRC press Boca Raton, FL.

Grimm, K. J., Kuhl, A. P., & Zhang, Z. (2013). Measurement models, estimation, and the study of change. *Structural Equation Modeling: A Multidisciplinary Journal*, *20*(3), 504–517. doi: https://doi.org/10.1080/10705511.2013.797837

Grimm, K. J., Steele, J. S., Ram, N., & Nesselroade, J. R. (2013). Exploratory latent growth models in the structural equation modeling framework. *Structural Equation Modeling: A Multidisciplinary Journal*, *20*(4), 568–591. doi: https://doi.org/10.1080/10705511.2013.824775

Guo, R., Zhu, H., Chow, S.-M., & Ibrahim, J. G. (2012). Bayesian lasso for semiparametric structural equation models. *Biometrics*, *68*(2), 567–577. doi: https://doi.org/10.1111/j.1541-0420.2012.01751.x

Holzinger, K. J., & Swineford, F. (1939). A study in factor analysis: the stability of a bi-factor solution. *Supplementary Educational Monographs*.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. In *Proceedings of the royal society of london a: Mathematical, physical and engineering sciences* (Vol. 186, pp. 453–461).

Jeffreys, H. (1961). *Theory of probability*. Jeffreys, H.: Oxford University Press.

Jöreskog, K. G. (1969). A general approach to confirmatory maximum likelihood factor analysis. *Psychometrika*, *34*(2), 183 - 202. doi: https://doi.org/10.1007/BF02289343

Jöreskog, K. G. (1978). Structural analysis of covariance and correlation matrices. *Psychometrika*, *43*, 443–477. doi: https://doi.org/10.1007/BF02293808

Jöreskog, K. G., & Sörbom, D. (1993). *Lisrel 8: Structural equation modeling with the simplis command language*. Scientific Software International.

Kruschke, J. K. (2011). Bayesian assessment of null values via parameter estimation and model comparison. *Perspectives on Psychological Science*, *6*(3), 299–312. doi: https://doi.org/10.1177/1745691611406925

Lee, S.-Y. (2007). *Structure equation modeling: A bayesian approach*. John Wiley and Sons.

Lee, S.-Y., & Song, X.-Y. (2012). *Basic and advanced bayesian structural equation modeling: With applications in the medical and behavioral sciences*. John Wiley & Sons.

Liu, H., Depaoli, S., & Marvin, L. (2022). Understanding the deviance information criterion for sem: Cautions in prior specification. *Structural Equation Modeling: A Multidisciplinary Journal*, *29*(2), 278–294. doi: https://doi.org/10.1080/10705511.2021.1994407

Liu, H., Zhang, Z., & Grimm, K. J. (2016). Comparison of inverse wishart and separation-strategy priors for bayesian estimation of covariance parameter matrix in growth curve analysis. *Structural Equation Modeling: A Multidisciplinary Journal*, *23*(3), 354–367. doi: https://doi.org/10.1080/10705511.2015.1057285

Lu, Z., Zhang, Z., & Lubke, G. (2011). Bayesian inference for growth mixture models with latent class dependent missing data. *Multivariate behavioral research*, *46*(4), 567–597. doi: https://doi.org/10.1080/00273171.2011.589261

Lu, Z.-H., Chow, S.-M., & Loken, E. (2016). Bayesian factor analysis as a variable-selection problem: Alternative priors and consequences. *Multivariate Behavioral Research*, *51*(4), 519–539. doi: https://doi.org/10.1080/00273171.2016.1168279

MacCallum, R. C., & Austin, J. T. (2000). Applications of structural equation modeling in psychological research. *Annual review of psychology*, *51*(1), 201–226. doi: https://doi.org/10.1146/annurev.psych.51.1.201

MacCallum, R. C., Edwards, M. C., & Cai, L. (2012). Hopes and cautions in implementing bayesian structural equation modeling. *Psychological Methods*, *17*(3), 340-345. doi: https://doi.org/10.1037/a0027131

McArdle, J., & Nesselroade, J. (2003). Growth curve analysis in contemporary psychological research. *Comprehensive handbook of psychology: Research methods in psychology*, *2*, 447 - 480. doi: https://doi.org/10.1002/0471264385.wei021

Merkle, E. C., & Rosseel, Y. (2018). blavaan: Bayesian structural equation models via parameter expansion. *Journal of Statistical Software*, *85*(4), 340–345. doi: https://doi.org/10.18637/jss.v085.i04

Muthen, B., & Asparouhov, T. (2012). Bayesian structural equation modeling: A more flexible representation of substantive theory. *Psychological Methods*, *17*(3), 313-335. doi: https://doi.org/10.1037/a0026802

Palomo, J., Dunson, D., & Bollen, K. (2007). Bayesian structural equation modeling. *Handbook of latent variable and related models*, *163-188*. doi: https://doi.org/10.1016/B978-044452044-9/50011-2

Pan, J. H., Song, X. Y., Lee, S. Y., & Kwok, T. (2008). Longitudinal analysis of quality of life for stroke survivors using latent curve models. *Stroke*, *39*(10), 2795–2802. doi: https://doi.org/10.1161/STROKEAHA.108.515460

Rosseel, Y., Oberski, D., Byrnes, J., Vanbrabant, L., & Savalei, V. (2013). lavaan: Latent variable analysis.[software]. *URL http://CRAN. R-project. org/package= lavaan (R package version 0.5-14)*.

Rubin, D. B. (1987). The calculation of posterior distributions by data augmentation: Comment: A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The sir algorithm. *Journal of the American Statistical Association*, *82*(398), 543–546. doi: https://doi.org/10.2307/2289457

Smid, S. C., McNeish, D., Miočević, M., & van de Schoot, R. (2019).

Bayesian versus frequentist estimation for structural equation models in small sample contexts: A systematic review *Structural Equation Modeling: A Multidisciplinary Journal*, *27*(1), 131–161. doi: https://doi.org/10.1080/10705511.2019.1577140

Song, X.-Y., & Lu, Z.-H. (2010). Semiparametric latent variable models with bayesian p-splines. *Journal of Computational and Graphical Statistics*, *19*(3), 590–608. doi: https://doi.org/10.1198/jcgs.2010.09094

Tanner, M. A., & Wong, W. H. (1987). The calculation of posterior distributions by data augmentation. *Journal of the American statistical Association*, *82*(398), 528–540. doi: https://doi.org/10.2307/2289457

Van de Schoot, R., Winter, S. D., Ryan, O., Zondervan-Zwijnenburg, M., & Depaoli, S. (2017). A systematic review of bayesian articles in psychology: The last 25 years. *Psychological Methods*, *22*(2), 217–239. doi: https://doi.org/10.1037/met0000100

Van Dyk, D. A., & Meng, X.-L. (2001). The art of data augmentation. *Journal of Computational and Graphical Statistics*, *10*(1), 1–50. doi: https://doi.org/10.1198/10618600152418584

Villegas, C. (1969). On the a priori distribution of the covariance matrix. *The Annals of Mathematical Statistics*, *40*, 1098-1099. doi: https://doi.org/10.1214/aoms/1177697617

Wang, Y., Feng, X.-N., & Song, X.-Y. (2016). Bayesian quantile structural equation models. *Structural Equation Modeling: A Multidisciplinary Journal*, *23*(2), 246–258. doi: https://doi.org/10.1080/10705511.2015.1033057

Yuan, K.-H., Kouros, C., & Kelley, K. (2008). Diagnosis for covariance structure models by analyzing the path. *Structural Equation Modeling*, *15*, 564-602. doi: https://doi.org/10.1080/10705510802338991

Zhang, Z. (2021). A note on wishart and inverse wishart priors for covariance matrix. *Journal of Behavioral Data Science*, *1*(2), 119–126. doi: https://doi.org/https://doi.org/10.35566/jbds/v1n2/p2

Zhang, Z., Hamagami, F., Lijuan Wang, L., Nesselroade, J. R., & Grimm, K. J. (2007). Bayesian analysis of longitudinal data using growth curve models. *International Journal of Behavioral Development*, *31*(4), 374–383. doi: https://doi.org/10.1177/0165025407077764

Zhang, Z., Lai, K., Lu, Z., & Tong, X. (2013). Bayesian inference and application of robust growth curve models using student's t distribution. *Structural Equation Modeling: A Multidisciplinary Journal*, *20*(1), 47–78. doi: https://doi.org/10.1080/10705511.2013.742382

Zhang, Z., & Wang, L. (2012). A note on the robustness of a full bayesian method for nonignorable missing data analysis. *Brazilian Journal of Probability and Statistics*, *26*(3), 244–264. doi: https://doi.org/10.1214/10-BJPS132

Journal of Behavioral Data Science, 2022, *2* (2), 47–72.
DOI: `https://doi.org/10.35566/jbds/v2n2/p3`

The Performance of Gelman-Rubin and Geweke's Convergence Diagnostics of Monte Carlo Markov Chains in Bayesian Analysis

Han Du[1], Zijun Ke[2], Ge Jiang[3], and Sijia Huang[4]

[1] Department of Psychology, University of California, Los Angeles, USA
`hdu@psych.ucla.edu`
[2] Department of Psychology, Sun Yat-sen University, China
[3] Department of Educational Psychology, University of Illinois Urbana-Champaign, USA
[4] Counseling and Educational Psychology Department, Indiana University Bloomington, USA

Abstract. Bayesian statistics have been widely used given the development of Markov chain Monte Carlo sampling techniques and the growth of computational power. A major challenge of Bayesian methods that has not yet been fully addressed is how we can appropriately evaluate the convergence of the random samples to the target posterior distributions. In this paper, we focus on Gelman and Rubin's diagnostic (PSRF), Brooks and Gleman's diagnostic (MPSRF), and Geweke's diagnostics, and compare the Type I error rate and Type II error rate of seven convergence criteria: $MPSRF > 1.1$, any upper bound of PSRF is larger than 1.1, more than 5% of the upper bounds of PSRFs are larger than 1.1, any PSRF is larger than 1.1, more than 5% of PSRFs are larger than 1.1, any Geweke test statistic is larger than 1.96 or smaller than -1.96, and more than 5% of Geweke test statistics are larger than 1.96 or smaller than -1.96. Based on the simulation results, we recommend the upper bound of PSRF if we only can choose one diagnostic. When the number of estimated parameters is large, between the diagnostic per parameter (i.e., PSRF) or the multivariate diagnostic (i.e., MPSRF), we recommend the upper bound of PSRF over MPSRF. Additionally, we do not suggest claiming convergence at the analysis level while allowing a small proportion of the parameters to have significant convergence diagnosis results.

Keywords: Convergence diagnostics · Bayesian analysis · Gelman-Rubin diagnostic · Geweke diagnostic

In recent decades, Bayesian statistics have been widely used given the development of Markov chain Monte Carlo (MCMC) sampling techniques and the growth of computational power (e.g., Van de Schoot et al., 2017). They have been used in cognitive psychology (e.g., Lee, 2008), developmental psychology (e.g., Van de Schoot et al., 2014; Walker et al., 2007), social psychology (e.g., Marsman et al., 2017), and many other areas. In the Bayesian framework, parameters are treated as random variables. Thus, we need to specify prior distributions for unknown parameters and obtain their posterior distributions. A major challenge of Bayesian methods that has not yet been fully addressed is how we can appropriately evaluate the convergence of the random samples to the target posterior distributions and the convergence of posterior means to the target mean. With nonconverged results, researchers may obtain severely biased parameter estimates and misleading statistical inferences. Therefore, there is a critical need to develop keen diagnostic methods for appropriately assessing convergence.

In the Bayesian framework, an MCMC algorithm converges when it samples thoroughly and stably from a density. More specifically, a converged Markov chain should have two properties: *stationarity* and *mixing*. To claim convergence, Markov chains need to move around in the posterior density in an appropriate manner and to mix well throughout the support of the density. In other words, when there are multiple Markov chains supporting the same estimation, they should trace out a common distribution (Gelman et al., 2014).

Practically, the convergence of MCMC algorithms can be assessed by visual inspection (i.e., trace plots) as well as quantitative evaluation. Various quantitative methods have been proposed for assessing convergence. To name a few, there are Garren and Smith (2000), Gelman and Rubin (1992), Geweke (1992), Heidelberger and Welch (1983), Johnson (1996), Liu, Liu, and Rubin (1992), and Raftery and Lewis (1992). Among them, the Gelman and Rubin's method and Geweke 's method currently are the most commonly used diagnostics and are implemented in popular software and R packages. For example, Mplus (Muthén and Muthén, 2017), CODA (Plummer et al., 2015), and BUGS (Spiegelhalter et al., 1996) can implement the Gelman and Rubin's diagnostic, and CODA can implement the Geweke 's diagnostic. The Gelman and Rubin's diagnostic requires multiple MCMC chains with different starting values, and potential scale reduction factor (PSRF) is used for assessing the convergence of chains for individual parameters. Brooks and Gelman (1998) further generalized the univariate PSRF to a multivariate scale reduction factor (MPSRF), which tests all the parameters' convergences as a group. The Geweke's diagnostic only requires one MCMC chain, therefore it is generally less time-consuming in calculation.

Several papers (Brooks and Roberts, 1998; Cowles and Carlin, 1996; El Adlouni et al., 2006) reviewed and/or compared different convergence diagnostics with hypothetical examples. The common conclusion from these papers is that no method can perform well in all cases, therefore they recommended a joint use of all diagnostics. The recommendation is constructive in ensuring convergence, but it may be overly conservative and infeasible in practice. First, there are more than 10 convergence diagnostics, not to mention that those diagnostics require the analyses in multiple software. Researchers rarely perform all diagnostics in one real data analysis. Second, the statistical performance (i.e., Type I error rate which is the probability of rejecting a true null hypothesis that assumes convergence, and Type II error rate which is the probability of not rejecting a false null hypothesis that assumes convergence) of the Gelman and Rubin's and the Geweke's methods has not yet been evaluated in simulation studies. Additionally, the performances of Gelman and Rubin's and Geweke's diagnostics were examined in relatively complex models, such as bimodal mixture of trivariate normals (Cowles and Carlin, 1996) and shifting level model (El Adlouni et al., 2006), in which the analytical forms were unknown or hard to access. As a consequence, the performance of these diagnostics when convergence is ensured (e.g., Type I error rates) is still unknown.

Besides the challenge of having too many convergence diagnostics, another challenge is that there are usually multiple parameters in one analysis. If we assess the convergence of Markov chains of each parameter, we face a multiple testing problem for the entire analysis. If no correction is applied and we claim that the convergence for the entire analysis is achieved when the convergence assessment for every parameter is passed, the Type I error rate at analysis level (i.e., analysis-wise Type I error rate) can be substantially inflated. For example, suppose that there are 20 independent parameters and we use the Geweke's diagnostic where the Type I error rate per parameter is supposed to be 5%. The analysis-wise Type I error rate is then $1 - 0.95^{20} = 0.642$, which is far above the intended level (i.e., 0.05) and implies that it is too easy to obtain a non-convergence conclusion using the Geweke's diagnostic. Applying conventional multiple testing corrections such as the Bonferroni correction might help reduce the inflated Type I error rates. However, parameters usually are not independent, and as illustrated later, the cutoff value for the Geweke's diagnostic is approximated, therefore the actual performance of multiple testing corrections remains an open question. In terms of the Gelman and Rubin's diagnostic, its cutoff comes from researchers' recommendation. The Type I error rate of the Gelman and Rubin's method at the parameter level or the analysis level in the literature remains largely unknown.

Given the above-mentioned unanswered questions, we focus on Gelman and Rubin's diagnostic, Brooks and Gleman's multivariate diagnostic, and Geweke's diagnostic, and aim to answer the following three questions in this paper:

(1) If we only choose one diagnostic, which one should we adopt? Even if no method performs well in all conditions, we would like to select the relatively better one. Type I error rate and Type II error rate are the two frequently used criteria for evaluating the performance of an analytic method. We therefore investigate this question based on these two criteria.

(2) In high dimension cases (i.e., the number of parameters is large), should we rely on the diagnostic at the parameter level (i.e., PSRF) or at the analysis level (i.e., MPSRF)? Complex models with large numbers of parameters are not uncommon in real psychological studies. For example, structural equation modeling and latent space modeling can easily estimate 20, 50, or even more than 100 parameters.

(3) If we rely on the diagnostic at the parameter level, should we allow a small proportion of the parameters (e.g., 5%) to have significant convergence test results but still claim convergence at the analysis level?

The outline of this paper is as follows. In the "Convergence Diagnostics" section, an overview of Gelman and Rubin's diagnostic, Brooks and Gleman's multivariate diagnostic, and Geweke's diagnostic is given. In the "Simulation Study" section, we evaluate and compare the performance of seven convergence criteria from the three diagnostics in conditions with converged and nonconverged MCMC chains. In this way, the Type I error rates (when converged Markov chains are used) and the Type II error rates (when nonconverged Markov chains are used) of the seven criteria are evaluated. We end the paper with some concluding remarks in the "Conclusion" section.

1 Convergence Diagnostics

1.1 Gelman and Rubin's Diagnostic

Gelman and Rubin (1992) proposed a general approach that utilizes multiple Markov chains with different starting values to monitor the convergence of MCMC samples. This method compares variance within and across chains, which is similar to Analysis of Variance (ANOVA). Let θ_{ij} denote the ith iteration of parameter θ from the jth chain. First, we estimate the averaged within chain variance by

$$W = \frac{1}{m\,(n-1)}\ \overset{[}{j}= 1]m\sum\ \overset{[}{i}= 1]n\sum \left(\theta_{ij} - \bar{\theta}_j\right)^2,$$

where n is the number of iterations within each chain, m is the number of chains, and $\bar{\theta}_j = \frac{1}{n}\ \overset{[}{i}= 1]n\sum\theta_{ij}$ is the within chain mean. Second, we estimate the between chain variance as

$$B = \frac{n}{m-1}\ \overset{[}{j}= 1]m\sum \left(\bar{\theta}_j - \bar{\theta}\right)^2.$$

where $\bar{\theta} = \frac{1}{m}\ \overset{[}{j}= 1]m\sum\bar{\theta}_j$ is the grand mean over all iterations and all chains. Then, we compute the pooled variance estimate ($\hat{V}$), which is constructed as a weighted average of the between (B) and within chain variance estimates (W),

$$\hat{V} = \frac{(n-1)}{n}W + \left(1 + \frac{1}{m}\right)\frac{B}{n}. \tag{1}$$

The ratio of the pooled and within-chain estimators is

$$\hat{R} = \frac{\hat{V}}{W}.$$

If the m chains mix well and stay stationary, the pooled variance estimate and within-chain variance estimate should be close to each other, and $\hat{R}$ should be close to 1.

Since there exists a sampling error in the variance estimate $\hat{V}$, one can adjust $\hat{R}$ by multiplying $\hat{R}$ with a correction term. Brooks and Gelman (1998) calculated the correction term as $d/(d-2)$, where d is the estimated degrees of freedom for a student t distribution approximation to the sample distribution of $\hat{V}/V$. The corrected ratio is

$$\hat{R}^c = \frac{d}{d-2}\frac{\hat{V}}{W}.$$

Gelman and Rubin (1992) named the corrected ratio as potential scale reduction factor (PSRF). When the PSRF is large, Gelman and Rubin (1992) suggested that one can reduce the $\hat{V}$ or increase W by running longer Markov chains to better fully explore the target distribution. From the algorithm, it is clear that Gelman and Rubin's diagnostic focuses on testing mixing rather than not stationary.

We need a criterion to define how close PSRF to 1 is acceptable. Gelman and Rubin (1992) and Cowles and Carlin (1996) looked at the 97.5% quantiles (i.e., upper bound) of PSRF. In practice, researchers usually treat the upper bound of PSRF less than 1.1 as an indicator of convergence. Gelman and Rubin (1992) found that $\hat{R}$ is overestimated, therefore either PSRF or the upper bound of PSRF should be conservative. To the best of our knowledge, there is no mathematical investigation about whether PSRF or the upper bound of PSRF should be used and whether the cutoff should be 1.1. Using the upper bound of PSRF and a cutoff of 1.1 are practical guidelines established by researchers' experience. Some software provides the upper bound of PSRF and PSRF (e.g., Mplus, Muthén and Muthén, 2017; CODA, (Plummer et al., 2015); and BUGS, Spiegelhalter et al., 1996) and some software only provide PSRF (e.g., Stan, Carpenter et al., 2017).

There are three major criticisms of the Gelman and Rubin's diagnostic. First, the test relies on over-dispersed starting values. If the starting values are too close to each other in the target distribution, the multiple chains may perform similarly and mix well even when the model is impossible to converge (i.e., the model is not identified). Second, the Gelman and Rubin's diagnostic only considers the first two moments, mean and variance. When the posterior distribution is non-normal, the higher order moments (e.g., skewness and kurtosis) also provide information in summarizing the distribution, but these moments are ignored in the Gelman and Rubin's diagnostic. Third, Gelman and Rubin (1992) and Brooks and Gelman (1998) emphasized that they do not suggest only monitoring the parameters of interest, but suggested simultaneously monitoring the convergence of all the parameters in a model. When the number of parameters is large, it is more challenging for all parameters to pass the PSRF cutoff simultaneously. It is also difficult to interpret the results when some parameters converge but some do not.

1.2 Brooks and Gleman's Multivariate Diagnostic

Brooks and Gelman (1998) generalized the Gelman and Rubin's diagnostic to consider multiple parameters simultaneously. We denote $\boldsymbol{\theta}_{ij}$ as a vector of parameters in the ith iteration of from the jth chain. The within chain and between chain variances of all parameters are quantified by a variance-covariance matrix. More specifically, the within chain variance-covariance matrix is

$$\boldsymbol{W} = \frac{1}{m\left(n-1\right)} \overset{[}{j}= 1]m\sum \overset{[}{i}= 1]n\sum \left(\boldsymbol{\theta}_{ij} - \bar{\boldsymbol{\theta}}_j\right)\left(\boldsymbol{\theta}_{ij} - \bar{\boldsymbol{\theta}}_j\right)', \tag{2}$$

where $\bar{\boldsymbol{\theta}}_j$ is the mean of vectors within the jth chain. The between chain variance-covariance matrix is calculated as

$$\boldsymbol{B} = \frac{n}{m-1} \overset{[}{j}= 1]m\sum \left(\bar{\boldsymbol{\theta}}_j - \bar{\boldsymbol{\theta}}\right)\left(\bar{\boldsymbol{\theta}}_j - \bar{\boldsymbol{\theta}}\right)'.$$

where $\bar{\boldsymbol{\theta}}$ is the grand mean vector. Similar to the univariate case, the pooled variance-covariance matrix $\hat{\boldsymbol{V}}$ is

$$\hat{\boldsymbol{V}} = \frac{(n-1)}{n}\boldsymbol{W} + \left(1 + \frac{1}{m}\right)\frac{\boldsymbol{B}}{n}.$$

The distance between $\hat{\boldsymbol{V}}$ and $\boldsymbol{W}$ is quantified as

$$\hat{R}^p = \frac{(n-1)}{n} + \left(1 + \frac{1}{m}\right)\lambda_1,$$

where λ_1 is the largest eigenvalue of $\boldsymbol{W}^{-1}\boldsymbol{B}/n$. Brooks and Gelman (1998) called $\hat{R}^p$ the multivariate PSRF (or MPSRF). MPSRF should approach 1 when convergence is achieved. Brooks and Gelman (1998) proved that MPSRF was an upper bound of the largest PSRF of all parameters.

The primary advantage of Brooks and Gleman's multivariate diagnostic is that MPSRF summarizes the PSRF sequences as a single value therefore it is easier to interpret than PSRF. Additionally, it is more computationally efficient than the computing all the PSRF sequences. However, Brooks and Gelman (1998) suggested reporting both MPSRF and PSRFs for all parameters, which largely diminishes the advantages of the multivariate diagnostic. Additionally, unlike PSRF, consensus has not yet been reached on the appropriate cut-offs for MPSRF. It is also unclear how the upper bound of MPSRF can be analytically calculated. To the best of our knowledge, no statistical software currently provides the estimates of the upper bound. As a consequence, in practice, it is difficult for researchers to conclude convergence using the multivariate approach, given that there is no clear guideline.

1.3 Geweke's Diagnostic

MCMC processes are special cases of stationary time series. Hence, based on the spectral density for time series, Geweke (1992) proposed a spectral density convergence diagnostic. The idea of Geweke's diagnostic is that in a convergent chain, the measures of two subsequences should be the equal. Assume there are two subsequences for one parameter, $\{\theta_A\}$ and $\{\theta_B\}$. The Geweke's statistic is a Z-score: the difference between the two sample means from the two subsequences divided by its estimated standard error. Geweke (1992) proposed that when the chain is stationary, the means of two subsequences are equal and Geweke's statistic has an asymptotically standard normal distribution,

$$Z = \frac{\bar{\theta}_A - \bar{\theta}_B}{\sqrt{\frac{1}{n_A}\hat{S}_A + \frac{1}{n_B}\hat{S}_B}} \xrightarrow{d} N(0,1)$$

where $\bar{\theta}_A$ and $\bar{\theta}_B$ are the means of the two subsequences, $\hat{S}_A$ and $\hat{S}_B$ are the variances of the two subsequences, and n_A and n_B are the numbers of iterations of the two subsequences. The null hypothesis of equal location which indicates convergence is rejected when Z is large (i.e., $|Z| > 1.96$). From the algorithm, we can see that the Geweke's diagnostic focuses on testing stationary rather than mixing. One assumption underlying Geweke's diagnostic is that the two subsequences are asymptotically independent. Hence, Geweke (1992) suggested taking the first 10% and the last 50%. Brooks and Gelman (1998) stated that the choice of two subsequences was arbitrary, and no general guidelines were available. Same as PSRF, the Geweke's diagnostic is for each parameter, and there is no multivariate version of Geweke's diagnostic. Hence, with Geweke's diagnostic, it is challenging to ensure all parameters converge and it is difficult to interpret the results when only part of the parameters converge.

2 Simulation Study

To answer the three questions raised in the introduction section, we conducted five simulation studies to explore the performances of Gelman and Rubin's diagnostic (PSRF), Brooks and Gleman's Multivariate diagnostic (MPSRF), and Geweke's diagnostic when (1) convergence should not be an issue (the null hypothesis is true) and (2) the chains should not converge (the null hypothesis is false). In the first condition, to ensure that the null hypothesis was true, we drew parameters from their analytically derived marginal posterior distributions to ensure convergence. In this way, convergence could be guaranteed. More specifically, a regression model and a multivariate normal model were considered and the Type I error rates of the studied diagnostic methods were evaluated. In the second condition, to ensure that the null hypothesis was false and the generated Markov chains would not converge, we used unidentified models, given that unidentified models were not estimable and estimation algorithms to these models generally would not converge. We considered a factor analysis model and investigated Type II error rates in this condition. The simulation code is available at `https://github.com/hduquant/Convergence-Diagnostics.git`.

We considered seven criteria in checking convergence based on the three diagnostics: (1) whether MPSRF was larger than 1.1 ($MPSRF > 1.1$), (2) whether any upper bound of PSRF was larger than 1.1 ($PSRF_{upper} > 1.1$), (3) whether more than 5% of all parameters' the upper bounds of PSRFs were larger than 1.1 ($PSRF_{upper,5\%} > 1.1$), (4) whether any PSRF was larger than 1.1 ($PSRF > 1.1$), (5) whether more than 5% of all parameters' PSRFs were larger than 1.1 ($PSRF_{5\%} > 1.1$), (6) whether any Geweke test statistic was larger than 1.96 or smaller than -1.96 ($|Geweke| > 1.96$), (7) whether more than 5% of Geweke test statistics were larger than 1.96 or smaller than -1.96 ($|Geweke|_{5\%} > 1.96$). If the answer was yes, we concluded that the MCMC chains failed to converge. We used 1.1 as the cutoff for MPSRF because there was no specific guideline in the literature. To mimic the cutoff for the upper bound of PSRF, we adopted 1.1.

We considered $PSRF_{upper,5\%} > 1.1$, $PSRF_{5\%} > 1.1$, and $|Geweke|_{5\%} > 1.96$ because when there are a large amount of parameters to be estimated, we may increase our tolerance for "significant" results per analysis. Specifically, we claimed nonconvergence at the analysis level if more than 5% of the convergence assessments based on the PSRFs, the upper bounds of PSRFs, or the Geweke's diagnostic were found to yield "significant" results (i.e., $PSRF_{5\%} > 1.1$, $PSRF_{upper,5\%} > 1.1$, and $|Geweke|_{5\%} > 1.96$). For $PSRF_{upper} > 1.1$, $PSRF > 1.1$, and $|Geweke| > 1.96$, we concluded non-convergence as any upper bound of PSRF, any PSRF, or any Geweke's value was above its corresponding cutoff.

An ideal diagnostic method was expected to yield a rejection rate of 5% across replications when the null hypothesis was true. When the null hypothesis was false, the ideal diagnostic method was expected to correctly reject the null hypothesis as frequently as possible. That is, the Type II error rates were expected to be as small as possible. We used two MCMC chains with different starting values to calculate PSRF and MPSRF. Based on one of the chains, we calculated Geweke's diagnostic values.

2.1 Type I Error Rates: Regression

We considered a multiple regression model with N individuals and p predictors,

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{e},$$

where $\boldsymbol{e} \sim N\left(0, \boldsymbol{I}\sigma^2\right)$ and $\boldsymbol{X} \sim N\left(\boldsymbol{0}, \boldsymbol{I}\right)$. In the simulation, $\sigma^2 = 0.25$. The population intercept and slopes ($\boldsymbol{\beta}$) were all 1. We used the Jeffreys priors for the residual variance and each regression coefficient,

$$f(\boldsymbol{\beta}) \propto 1,$$
$$f(\sigma^2) \propto \left(\sigma^2\right)^{-1}.$$

Denote $\hat{\boldsymbol{\beta}} = \left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\boldsymbol{X}'\boldsymbol{y}$ and $\hat{\sigma^2} = \frac{1}{N-p-1}\left(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}\right)'\left(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\beta}}\right)$. The marginal posterior distribution of σ^2 is an inverse-Gamma distribution,

$$f(\sigma^2|\boldsymbol{X}, \boldsymbol{y}) = IG\left(\frac{N-p-1}{2}, \frac{(N-p-1)\,\hat{\sigma^2}}{2}\right).$$

The marginal posterior distribution of $\boldsymbol{\beta}$ is a multivariate student-t distribution,

$$f(\boldsymbol{\beta}|\boldsymbol{X}, \boldsymbol{y}) = t_{N-p-1}\left(\hat{\boldsymbol{\beta}}, \hat{\sigma^2}\left(\boldsymbol{X}'\boldsymbol{X}\right)^{-1}\right).$$

The number of parameters to be estimated was $p+2$ (p slopes, 1 intercept, and 1 residual variance). We varied the sample size N ($N = 100$, 200, 500, and 1000) and the number of predictors p ($p = 5$, 10, 50, 80, 90, and 100). The conditions of N were nested within p because N should be larger than p to ensure model identification. We calculated the seven criteria when the number of iterations (n) was 100, 500, 10^3, 3×10^3, 5×10^3, 10^4, 5×10^4, and 10^5. We report the proportions of rejecting the convergence (false rejection rates or empirical Type I Error rates) across 1000 replications of $PSRF_{upper} > 1.1$ and $PSRF_{upper,5\%} > 1.1$ in Table 1, the rejection rates of $PSRF > 1.1$, $PSRF_{5\%} > 1.1$, and $MPSRF > 1.1$ in Table 2, and the rejection rates of $|Geweke|_{5\%} > 1.96$ and $|Geweke| > 1.96$ in Table 3. We omit the columns of the number of iterations where all rejection rates are 0.

We summarized our findings as below and in Table 4. First, more iterations (i.e., larger n) helped reach convergence conclusions for all seven indices (see Tables 1-3). The rejection rates (empirical Type I error rates) generally decreased as the number of iterations increased. When the number of iterations was 100 and the number of predictors was 100, MPSRF even could not be calculated because $\boldsymbol{W}$ in Equation (2) was not positive definite. The rejection rates from the five indices based on PSRF and MPSRF went down to 0% instead of 5% as the number of iterations became larger. It is consistent with the conclusion from Gelman and Rubin (1992) that using the upper bound of PSRF or PSRF should be too conservative.

Second, whether allowing the upper bound of PSRF, PSRF, or Geweke's diagnostic to reject convergence by 5% of the parameters ($PSRF_{upper,5\%} > 1.1$, $PSRF_{5\%} > 1.1$, and $|Geweke|_{5\%} > 1.96$) in each analysis depended on the number of parameters. When the number of parameters (p) was smaller than 20, it was impossible to reject 5% of the parameters' convergences since $20 \times 5\% = 1$ and we could not reject < 1 number of parameters. Hence, when $p \leq 20$, there is no need to distinguish $PSRF_{upper,5\%} > 1.1$ vs. $PSRF_{upper} > 1.1$, $PSRF_{5\%} > 1.1$ vs. $PSRF > 1.1$, $|Geweke|_{5\%} > 1.96$ vs. $|Geweke| > 1.96$. In other words, $PSRF_{upper,5\%} > 1.1$ is equivalent to $PSRF_{upper} > 1.1$, $PSRF_{5\%} > 1.1$ is equivalent to $PSRF > 1.1$, $|Geweke|_{5\%} > 1.96$ is equivalent to $|Geweke| > 1.96$ (see Tables 1-3). When $p \geq 50$, as expected, allowing 5% significant results per dataset had lower rejection rates than not allowing any significant results per dataset. But this difference only appeared when the number of iterations was small. When the number of iterations was 1000, both the rejection rates from $PSRF_{upper,5\%} > 1.1$ and $PSRF_{upper} > 1.1$ were below 5% (see Table 1), and when the number of iterations was 500, both the rejection rates from $PSRF_{5\%} > 1.1$ and $PSRF > 1.1$ were all below 5% (see Table 2). Additionally, with $PSRF_{upper} > 1.1$, $PSRF > 1.1$, $MPSRF > 1.1$, and $|Geweke| > 1.96$, it was more difficult to reach the convergence conclusion with more parameters since we held a strict criteria by not allowing any significant results. But

Table 1: Empirical Type I Error Rates for $PSRF_{upper,5\%} > 1.1$ and $PSRF_{upper} > 1.1$ in the Regression Study

N	p	$PSRF_{upper,5\%} > 1.1$			$PSRF_{upper} > 1.1$				
		100	500	10^3	100	500	10^3	3×10^3	5×10^3
100	5	0.885	0.061	0	0.885	0.061	0	0	0
200	5	0.875	0.076	0.002	0.875	0.076	0.002	0	0
500	5	0.862	0.061	0.003	0.862	0.061	0.003	0	0
1000	5	0.867	0.066	0.002	0.867	0.066	0.002	0	0
100	10	0.966	0.104	0.003	0.966	0.104	0.003	0	0
200	10	0.965	0.099	0.005	0.965	0.099	0.005	0	0
500	10	0.973	0.116	0.006	0.973	0.116	0.006	0	0
1000	10	0.978	0.119	0.006	0.978	0.119	0.006	0	0
100	50	1	0.027	0	1.000	0.357	0.010	0	0
200	50	1	0.017	0	1.000	0.396	0.022	0	0
500	50	1	0.013	0	1.000	0.361	0.012	0	0
1000	50	1	0.011	0	1.000	0.368	0.009	0	0
100	80	0.999	0.024	0	1.000	0.453	0.023	0	0
200	80	1	0.004	0	1.000	0.540	0.023	0	0
500	80	1	0.003	0	1.000	0.526	0.023	0	0
1000	80	1	0	0	1.000	0.531	0.014	0	0
100	90	0.998	0.045	0.001	1.000	0.442	0.040	0.005	0.003
200	90	1	0.005	0	1.000	0.578	0.022	0	0
500	90	1	0.003	0	1.000	0.574	0.014	0	0
1000	90	1	0.005	0	1.000	0.581	0.026	0	0
200	100	-	0.006	0	-	0.595	0.030	0	0
500	100	-	0	0	-	0.610	0.021	0	0
1000	100	-	0.001	0	-	0.613	0.028	0	0

Note. "-" indicates that only few replications had results in that condition, therefore the Type I error rates were not reliable and thus not reported.

for $PSRF_{upper,5\%} > 1.1$ and $PSRF_{5\%} > 1.1$, it could be easier to reach the convergence conclusion with more parameters.

Third, not surprisingly, using PSRF to assess convergence was more conservative than using the upper bound of PSRF. With the same number of iterations, the rejection rates from PSRFs were lower than those from the upper bounds of PSRFs (see Tables 1 and 2). Fourth, MPSRF was sensitive to the number of estimated parameters. When $p \geq 80$, the rejection rates from $MPSRF > 1.1$ were high (e.g., 0.706) and 3000 iterations were needed to reduce the rejection rates below 5% (see Table 2). Fifth, the Geweke's convergence diagnostic, $|Geweke|_{5\%} > 1.96$ and $|Geweke| > 1.96$, tended to overestimate non-convergence. Even 10^5 iterations failed to reduce the rejection rates below 5%.

Table 2: Empirical Type I Error Rates for $PSRF_{5\%} > 1.1$, $PSRF > 1.1$, and $MPSRF > 1.1$ in the Regression Study

N	p	$PSRF_{5\%} > 1.1$	$PSRF > 1.1$					$MPSRF > 1.1$		
		100	100	500	10^3	3×10^3	5×10^3	100	500	10^3
100	5	0.167	0.167	0	0	0	0	0.262	0	0
200	5	0.143	0.143	0	0	0	0	0.218	0	0
500	5	0.148	0.148	0	0	0	0	0.243	0	0
1000	5	0.130	0.13	0	0	0	0	0.222	0	0
100	10	0.261	0.261	0	0	0	0	0.674	0	0
200	10	0.237	0.237	0	0	0	0	0.699	0	0
500	10	0.232	0.232	0	0	0	0	0.64	0	0
1000	10	0.217	0.217	0	0	0	0	0.655	0	0
100	50	0.124	0.604	0	0	0	0	1	0.638	0
200	50	0.111	0.649	0	0	0	0	1	0.654	0
500	50	0.123	0.664	0	0	0	0	1	0.647	0
1000	50	0.120	0.659	0	0	0	0	1	0.674	0
100	80	0.103	0.701	0	0	0	0	1	0.999	0.144
200	80	0.041	0.802	0	0	0	0	1	1	0.158
500	80	0.041	0.824	0	0	0	0	1	0.998	0.152
1000	80	0.037	0.844	0	0	0	0	1	1.000	0.133
100	90	0.187	0.781	0.038	0.009	0.003	0.003	1	1.000	0.372
200	90	0.070	0.845	0	0	0	0	1	1.000	0.413
500	90	0.058	0.852	0	0	0	0	1	1.000	0.375
1000	90	0.056	0.861	0	0	0	0	1	1.000	0.408
200	100	-	-	0	0	0	0	-	1.000	0.692
500	100	-	-	0	0	0	0	-	1.000	0.661
1000	100	-	-	0	0	0	0	-	1.000	0.706

Note. Same as Table 1.

Table 3: Empirical Type I Error Rates for $PSRF_{5\%} > 1.1$, $PSRF > 1.1$, and $MPSRF > 1.1$ in the Regression Study

N	p	$\lvert Geweke\rvert_{5\%} > 1.96$								$\lvert Geweke\rvert > 1.96$							
		100	500	10^3	3×10^3	5×10^3	10^4	5×10^4	10^5	100	500	10^3	3×10^3	5×10^3	10^4	5×10^4	10^5
100	5	0.515	0.396	0.363	0.319	0.321	0.304	0.273	0.275	0.515	0.396	0.363	0.319	0.321	0.304	0.273	0.275
200	5	0.503	0.389	0.363	0.336	0.305	0.321	0.281	0.311	0.503	0.389	0.363	0.336	0.305	0.321	0.281	0.311
500	5	0.517	0.394	0.330	0.318	0.304	0.330	0.309	0.306	0.517	0.394	0.330	0.318	0.304	0.330	0.309	0.306
1000	5	0.515	0.364	0.351	0.302	0.300	0.303	0.305	0.298	0.515	0.364	0.351	0.302	0.300	0.303	0.305	0.298
100	10	0.709	0.554	0.523	0.496	0.467	0.466	0.451	0.459	0.709	0.554	0.523	0.496	0.467	0.466	0.451	0.459
200	10	0.727	0.575	0.517	0.437	0.441	0.461	0.474	0.427	0.727	0.575	0.517	0.437	0.441	0.461	0.474	0.427
500	10	0.696	0.564	0.549	0.492	0.468	0.466	0.470	0.487	0.696	0.564	0.549	0.492	0.468	0.466	0.470	0.487
1000	10	0.715	0.554	0.528	0.466	0.478	0.483	0.482	0.433	0.715	0.554	0.528	0.466	0.478	0.483	0.482	0.433
100	50	0.846	0.635	0.564	0.526	0.469	0.457	0.460	0.471	0.981	0.950	0.928	0.903	0.891	0.879	0.875	0.891
200	50	0.884	0.698	0.581	0.522	0.503	0.474	0.466	0.488	0.993	0.968	0.936	0.933	0.926	0.910	0.916	0.908
500	50	0.883	0.698	0.607	0.556	0.508	0.505	0.486	0.500	0.990	0.970	0.957	0.937	0.941	0.924	0.931	0.931
1000	50	0.881	0.698	0.625	0.512	0.509	0.495	0.498	0.453	0.995	0.974	0.956	0.930	0.946	0.940	0.924	0.917
100	80	0.772	0.532	0.446	0.392	0.387	0.382	0.342	0.351	0.996	0.954	0.942	0.927	0.914	0.910	0.915	0.906
200	80	0.889	0.646	0.534	0.461	0.471	0.424	0.395	0.378	0.998	0.993	0.981	0.987	0.981	0.977	0.971	0.969
500	80	0.896	0.657	0.526	0.471	0.435	0.429	0.373	0.392	1.000	0.991	0.992	0.987	0.986	0.992	0.987	0.979
1000	80	0.899	0.644	0.549	0.451	0.424	0.388	0.409	0.395	1.000	0.998	0.993	0.990	0.992	0.983	0.980	0.985
100	90	0.722	0.490	0.446	0.417	0.391	0.369	0.356	0.393	0.985	0.950	0.910	0.887	0.859	0.872	0.851	0.871
200	90	0.928	0.714	0.621	0.525	0.537	0.489	0.484	0.482	1.000	0.995	0.987	0.979	0.981	0.985	0.979	0.977
500	90	0.940	0.750	0.628	0.558	0.567	0.498	0.477	0.480	1.000	1.000	0.994	0.997	0.987	0.990	0.991	0.991
1000	90	0.934	0.763	0.651	0.554	0.523	0.472	0.511	0.499	0.999	0.998	0.996	0.993	0.995	0.995	0.992	0.993
200	100	0.903	0.665	0.557	0.460	0.461	0.440	0.399	0.384	1.000	0.998	0.994	0.991	0.993	0.992	0.989	0.985
500	100	0.942	0.674	0.560	0.454	0.440	0.432	0.387	0.401	1.000	0.999	0.996	0.992	0.992	0.989	0.995	0.993
1000	100	0.933	0.703	0.590	0.458	0.458	0.417	0.410	0.390	1.000	1.000	0.998	0.996	0.994	0.994	0.992	0.993

Table 4: Summary of 7 indices in the regression model, multivariate normal model, and factor analysis model

	$PSRF_{upper,5\%}$ > 1.1	$PSRF_{upper}$ > 1.1	$PSRF_{5\%}$ > 1.1	$PSRF$ > 1.1	$MPSRF$ > 1.1	$\lvert Geweke\rvert_{5\%}$ > 1.96	$\lvert Geweke\rvert$ > 1.96
Regression							
Do more iterations improve convergence?	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Needed iterations to control Type I error rates below 5%	1000 when $p \leq 10$ 500 when $p \geq 50$	1000	500	500	3000	NA	NA
Do more parameters to be estimated make it more difficult to attain the convergence threshold?	Easier	More difficult	Easier	More difficult	More difficult	No pattern	More difficult
Multivariate normal							
Do more iterations improve convergence?	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Needed iterations to control Type I error rates below 5%	500	1000	500	500	3000	NA	NA
Do more parameters to be estimated make it more difficult to attain the convergence threshold?	Easier	More difficult	Easier	More difficult	More difficult	No pattern	More difficult
Factor Analysis							
Do more iterations increase Type II error rates?		Yes		Yes	Yes		Yes

2.2 Type I Error Rates: Multivariate Normal

We considered a multivariate normal model with N individuals and p variables ($\boldsymbol{x} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$). The population mean ($\boldsymbol{\mu}$) was a vector of 0 and the population covariance matrix ($\mathbf{\Sigma}$) had variances of 1 and covariances of 0.3. We used the Jeffreys priors for the mean and covariance matrix,

$$f(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \propto |\boldsymbol{\Sigma}|^{-(p+1)/2}.$$

The marginal posterior distribution of $\mathbf{\Sigma}$ was an inverse-Wishart distribution,

$$f(\boldsymbol{\Sigma}|\boldsymbol{x}) \sim IW(n-1, \boldsymbol{S}),$$

where $S = \sum_{i=1}^{n} (\boldsymbol{x_i} - \bar{\boldsymbol{x}})(\boldsymbol{x_i} - \bar{\boldsymbol{x}})'$ and $\bar{\boldsymbol{x}}$ was the sample mean. The marginal posterior distribution of $\boldsymbol{\mu}$ is a multivariate student-t distribution,

$$f(\boldsymbol{\mu}|\boldsymbol{x}) \sim \boldsymbol{t_{n-p}}(\bar{\boldsymbol{x}}, \boldsymbol{S}/(n(n-p))).$$

The number of parameters to be estimated was $p(p+1)/2 + p$ (i.e., p mean structure components and $p(p+1)/2$ variance-covariance structure components). We varied the sample size N (N = 100, 200, 500, and 1000) and the number of variables p (p = 5, 10, 12, and 20). Similar to the multiple regression case, the conditions of N were nested within p because N should be larger than p to ensure convergence. We calculated the seven criteria when the number of iterations (n) ranged from 100 to 10^5.

We report the proportions of rejecting the convergence (empirical Type I error rates) across 1000 replications of seven criteria in Table 5 and omit the columns of n where all rejection rates are 0. All findings were consistent with the findings in the regression case. We summarized the findings in Table 4. Since the number of parameters was relatively high in the multivariate normal case (e.g., when $p = 5$, the number of parameters was 20), the difference of the rejection rates between indices allowing 5% significant results in an analysis and indices not allowing any significant results was larger than the regression case. But again this difference only appeared when the number of iterations was small.

2.3 Type II Error Rates: Factor Analysis

We considered a confirmatory factor analysis model with one factor (also called latent variable) and five manifest variables. $\boldsymbol{x}$ was simulated from a confirmatory factor analysis (CFA) model

$$\boldsymbol{x} = \boldsymbol{\mu} + \mathbf{\Lambda}\boldsymbol{\xi} + \boldsymbol{\varepsilon}, \tag{3}$$

where $\boldsymbol{\mu}$ was a vector of 0, $\mathbf{\Lambda}$ was a 5×1 vector of factor loadings, $\boldsymbol{\xi}$ was a scalar of factor scores, and $\boldsymbol{\varepsilon}$ was a 5×1 vector of independent measurement errors for 5 manifest variables. Let $\mathbf{\Phi} = cov(\boldsymbol{\xi})$ and $\mathbf{\Psi} = cov(\boldsymbol{\varepsilon})$, then the corresponding population covariance matrix of $\boldsymbol{x}$ was $\mathbf{\Sigma} = \mathbf{\Lambda}\mathbf{\Phi}\mathbf{\Lambda}' + \mathbf{\Psi}$. In the

Table 5: Empirical Type I Error Rates for $PSRF_{upper,5\%} > 1.1$, $PSRF_{upper} > 1.1$, $PSRF_{5\%} > 1.1$, $PSRF > 1.1$, $MPSRF > 1.1$, $|Geweke|_{5\%} > 1.96$, and $|Geweke| > 1.96$ in the Multivariate Normal Study

N	p	$PSRF_{upper,5\%} > 1.1$		$PSRF_{upper} > 1.1$			$PSRF_{5\%} > 1.1$	$PSRF > 1.1$	$MPSRF > 1.1$			
		100	500	100	500	10^3	100	100	100	500	10^3	3×10^3
100	5	0.957	0.033	0.992	0.16	0	0.104	0.319	0.985	0	0	0
200	5	0.965	0.029	0.998	0.146	0.003	0.097	0.334	0.982	0.001	0	0
500	5	0.940	0.034	0.988	0.149	0.003	0.095	0.331	0.971	0	0	0
1000	5	0.948	0.027	0.991	0.150	0.007	0.084	0.338	0.976	0	0	0
100	10	0.998	0.036	1.000	0.411	0.008	0.134	0.682	1	0.954	0.007	0
200	10	1.000	0.018	1.000	0.374	0.015	0.108	0.683	1	0.961	0.011	0
500	10	0.998	0.028	1.000	0.396	0.015	0.102	0.676	1	0.955	0.012	0
1000	10	1.000	0.022	1.000	0.384	0.006	0.103	0.637	1	0.965	0.002	0
200	12	1.000	0.028	1.000	0.506	0.020	0.092	0.77	1	1	0.36	0
500	12	1.000	0.019	1.000	0.443	0.016	0.103	0.752	1	1	0.344	0
1000	12	1.000	0.026	1.000	0.461	0.013	0.095	0.764	1	1	0.32	0
500	20	-	0.012	-	0.745	0.044	-	-	-	1	1	0.009
1000	20	-	0.015	-	0.733	0.051	-	-	-	1	1	0.005
		$\lvert Geweke\rvert_{5\%} > 1.96$										
		100	500	10^3	3×10^3	5×10^3	10^4	5×10^4	10^5			
100	5	0.514	0.359	0.314	0.269	0.279	0.291	0.245	0.246			
200	5	0.560	0.376	0.327	0.291	0.275	0.242	0.250	0.230			
500	5	0.543	0.346	0.334	0.280	0.280	0.273	0.264	0.255			
1000	5	0.538	0.369	0.310	0.281	0.251	0.258	0.259	0.254			
100	10	0.764	0.544	0.453	0.402	0.393	0.359	0.395	0.356			
200	10	0.760	0.499	0.464	0.384	0.385	0.391	0.367	0.402			
500	10	0.787	0.536	0.495	0.412	0.407	0.387	0.358	0.384			
1000	10	0.777	0.555	0.476	0.391	0.390	0.348	0.370	0.363			
200	12	0.820	0.567	0.499	0.424	0.466	0.405	0.408	0.384			
500	12	0.819	0.613	0.527	0.440	0.394	0.411	0.417	0.411			
1000	12	0.835	0.620	0.523	0.454	0.448	0.395	0.372	0.381			
500	20	0.905	0.631	0.500	0.453	0.393	0.410	0.383	0.376			
1000	20	0.911	0.642	0.528	0.460	0.420	0.410	0.381	0.352			
		$\lvert Geweke\rvert > 1.96$										
		100	500	10^3	3×10^3	5×10^3	10^4	5×10^4	10^5			
100	5	0.808	0.693	0.620	0.588	0.606	0.592	0.555	0.566			
200	5	0.834	0.693	0.658	0.629	0.593	0.578	0.532	0.564			
500	5	0.818	0.682	0.652	0.600	0.586	0.599	0.574	0.566			
1000	5	0.814	0.691	0.635	0.592	0.577	0.579	0.561	0.554			
100	10	0.991	0.963	0.936	0.913	0.898	0.892	0.903	0.893			
200	10	0.991	0.959	0.937	0.895	0.893	0.908	0.893	0.885			
500	10	0.988	0.956	0.944	0.904	0.913	0.896	0.899	0.902			
1000	10	0.992	0.968	0.947	0.918	0.896	0.888	0.891	0.901			
200	12	0.996	0.984	0.978	0.951	0.951	0.951	0.949	0.951			
500	12	0.999	0.989	0.981	0.967	0.948	0.950	0.967	0.956			
1000	12	0.997	0.989	0.984	0.965	0.953	0.966	0.938	0.950			
500	20	1.000	0.998	0.999	0.999	0.998	0.999	0.999	1.000			
1000	20	1.000	0.999	1.000	1.000	1.000	0.998	0.997	0.998			

Note. Same as Table 1.

simulation, $\mathbf{\Lambda} = (0.75, 0.75, 0.8, 0.85, 0.95)'$ and $\boldsymbol{\xi} \sim N(0,1)$. $\mathbf{\Psi}$ was calculated to ensure that the diagonal elements of $\mathbf{\Sigma}$ were 1.

Because latent variables are unobserved, their measurement units must be specified by researchers. There are two ways to fix the measurement units of latent variables. The first way is that for each latent variable, one of the corresponding manifest variables should have a factor loading of 1. The alternative option is that the variances of all latent variables are fixed at 1. We aimed to create a condition where the model is not identified and thus the MCMC chains should not converge, and thus we did not put any constraints. We freely estimated all of the 5 factor loadings and the factor variance. Hence, there were 11 parameters to be estimated (5 factor loadings, 5 residual variances, and 1 factor variance). Besides non-identification issue due to freely estimating all parameters, sign reflection invariance can also cause non-identification. Sign reflection invariance refers to a phenomenon where the signs of factor loadings and their associated factors change simultaneously while the model fit remains the same (Erosheva and Curtis, 2017). In the Bayesian framework, sign reflection invariance may result in multimodality in posterior distributions and cause non-convergence. A typical solution is to place positivity constraints on the priors of loadings to ensure that a loading per factor is positive. We considered both the positivity constraint case which could avoid sign reflection invariance and the case without a positivity constraint where non-identification was due to both freely estimating all parameters and sign reflection invariance.

We considered several widely used noninformative priors: $\mathbf{\Phi} \sim IG\,(0.001, 0.001)$, each diagonal element in $\mathbf{\Psi}$ followed $IG\,(0.001, 0.001)$, and each element in $\mathbf{\Lambda}$ followed $N\left(0, 10^6\right)$ without a positivity constraint or followed $Uniform\left(0, 10^6\right)$ with a positivity constraint. There were no analytical forms for the posterior distribution. Hence, we used the Gibbs sampling algorithm to estimate the variables one at a time in a sequence (Gelfand and Smith, 1990) and the Metropolis-Hastings algorithm (Gilks et al., 1996; Hastings, 1970) to empirically construct the posterior distributions. In Gibbs sampling and Metropolis-Hastings algorithm, a number of early iterations before convergence should be discarded (i.e., burn-in period) since they are not representative samples of the target distribution (Gelman et al., 2014; Lynch, 2007). Although in our case the MCMC chains should not converge regardless of the length of burn-in period, we still discarded the first 1000 iterations and used the first half of the left chain as a second burn-in period. The sample size N varied as 100, 200, 500, or 1000. With 11 parameters, it was impossible to reject the null that assumed convergence for 5% of the parameters, because $11 * 5\% < 1$. Therefore, we did not consider $PSRF_{upper,5\%} > 1.1$, $PSRF_{5\%} > 1.1$, and $|Geweke|_{5\%} > 1.96$ in this condition. When any upper bound of PSRF was larger than 1.1 ($PSRF_{upper} > 1.1$), any PSRF was larger than 1.1 ($PSRF > 1.1$), or any absolute value of the Geweke's diagnostic was larger than 1.96 ($|Geweke| > 1.96$), we concluded non-convergence. We calculated the Type II error rates of $PSRF_{upper} > 1.1$, $PSRF > 1.1$, $MPSRF > 1.1$, and $|Geweke| > 1.96$ when the number of iterations after the two burn-in periods was from 250 to 5×10^4. Note that even un-identified model can still generate converged results by coincidence. Hence, the Type II error rates indeed are generally underestimated.

Positivity Constraint We first focus on the case with positivity constraints on factor loadings. We report the false acceptance rates (the empirical Type II error rates) across 1000 replications of the four indices in Table 6. As shown in Table 6, as the sample size (N) decreased, the Type II error rates increased in all four indices. This is because as the sample size decreased, the amount of information in the data became smaller compared to that in the prior. Consequently, Bayesian methods increasingly relied on the prior, which was stationary per se. The stationary prior would make the resulting Markov chains appear to be stationary when prior was heavily weighted.

Table 6: Empirical Type II Error Rates for $PSRF_{upper} > 1.1$, $PSRF > 1.1$, $MPSRF > 1.1$, and $|Geweke| > 1.96$ in the Factor Analysis Study with a Positivity Constraint

		$PSRF_{upper} > 1.1$						
N	n	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.001	0.003	0.011	0.023	0.037	0.129	0.223
100		0.003	0.002	0.003	0.006	0.011	0.037	0.082
200		0	0.001	0.002	0.003	0.002	0.018	0.019
500		0	0	0.002	0.001	0.001	0.001	0.003
1000		0	0	0.001	0.001	0	0	0
		$PSRF > 1.1$						
N	n	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.010	0.015	0.048	0.074	0.091	0.258	0.374
100		0.008	0.014	0.023	0.020	0.039	0.099	0.192
200		0.001	0.003	0.009	0.011	0.014	0.038	0.043
500		0.002	0.002	0.005	0.005	0.004	0.011	0.014
1000		0	0	0.003	0.001	0.001	0.002	0.001
		$MPSRF > 1.1$						
N	n	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.011	0.021	0.075	0.143	0.210	0.583	0.730
100		0.005	0.013	0.043	0.039	0.077	0.269	0.498
200		0.001	0.006	0.012	0.015	0.024	0.077	0.131
500		0	0.003	0.006	0.008	0.01	0.017	0.029
1000		0	0.001	0.005	0.003	0.001	0.003	0.002
		$\|Geweke\| > 1.96$						
N	n	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.059	0.116	0.095	0.096	0.188	0.260	0.343
100		0.068	0.105	0.071	0.136	0.112	0.190	0.174
200		0.066	0.091	0.119	0.082	0.046	0.063	0.096
500		0.117	0.094	0.134	0.128	0.136	0.098	0.053
1000		0.107	0.142	0.167	0.152	0.156	0.117	0.094

Besides quantitative diagnostic methods, the trace plot method could provide another piece of information. Figure 1 presents the trace plots of the 11 parameters when $N = 50$ with 10^5 iterations after the initial burn-in period. In this replication, different indices gave inconsistent conclusion: the upper bound of RSPF of the factor variance was 1.32, but the MRSPF was 1.07 and the absolute values of Geweke of all parameters were within 1.15. The trace plots for the factor variance and the 5th residual variance in Figure 1 showed several high peaks and were somewhat truncated at 0. This pattern is suspicious, which may be the signal of non-convergence, although MRSPF and Geweke's diagnostic indicated convergence in this case. Thus, we suggest using both quantitative diagnostics and visual inspection in practice. When the sample size N became large, the prior information could not bound the posterior distributions. As shown in Figure 2, when $N = 1000$, the posterior samples of factor loadings kept going up and the two chains did not mix, and the posterior samples of factor variances were almost fixed at 0. In this case, the PSRFs (and their upper bounds) for factor loadings and factor variance and MPSRF were above 1.1, and the several absolute values of the Geweke's diagnostic (especially for the factor loadings and factor variance) were larger than 1.96.

We summarize other conclusions as below. First, for $PSRF_{upper} > 1.1$, $PSRF > 1.1$, and $MPSRF > 1.1$, more iterations made reaching convergence conclusions easier and thus increased the Type II error rates. More iterations increased the Type II error rates for $|Geweke| > 1.96$ when N was small. Second, because using PSRF is less conservative than using the upper bound, $PSRF > 1.1$ had larger Type II error rates compared to $PSRF_{upper} > 1.1$. $MPSRF > 1.1$ also had larger Type II error rates than $PSRF_{upper} > 1.1$. Third, with the positivity constraint, $|Geweke| > 1.96$ had similar Type II error rates as $PSRF > 1.1$.

No Positivity Constraint Now we move to the case without a positivity constraint on factor loadings where sign reflection invariance could cause non-convergence. We report the Type II error rates without a positivity constraint across 1000 replications of the four indices in Table 7. To better illustrate the posterior samples, Figure 3 presents the trace plots of the 11 parameters when $N = 1000$ without a positivity constraint. The two chains of factor loadings did not mix well. PSRF and MPSRF can test whether multiple chains mix well, whereas Geweke's test cannot test this feature of the posterior samples. As a consequence, the Type II error rates of $|Geweke| > 1.96$ were much larger than $PSRF_{upper} > 1.1$, $PSRF > 1.1$, and $MPSRF > 1.1$, which was different from the positivity constraint case. Similar to the positivity constraint case, $PSRF > 1.1$ and $MPSRF > 1.1$ had larger Type II error rates than $PSRF_{upper} > 1.1$.

3 Conclusion

Bayesian statistics has grown vastly and has been widely used in psychological studies in recent decades given the surge in computational power. Conver-

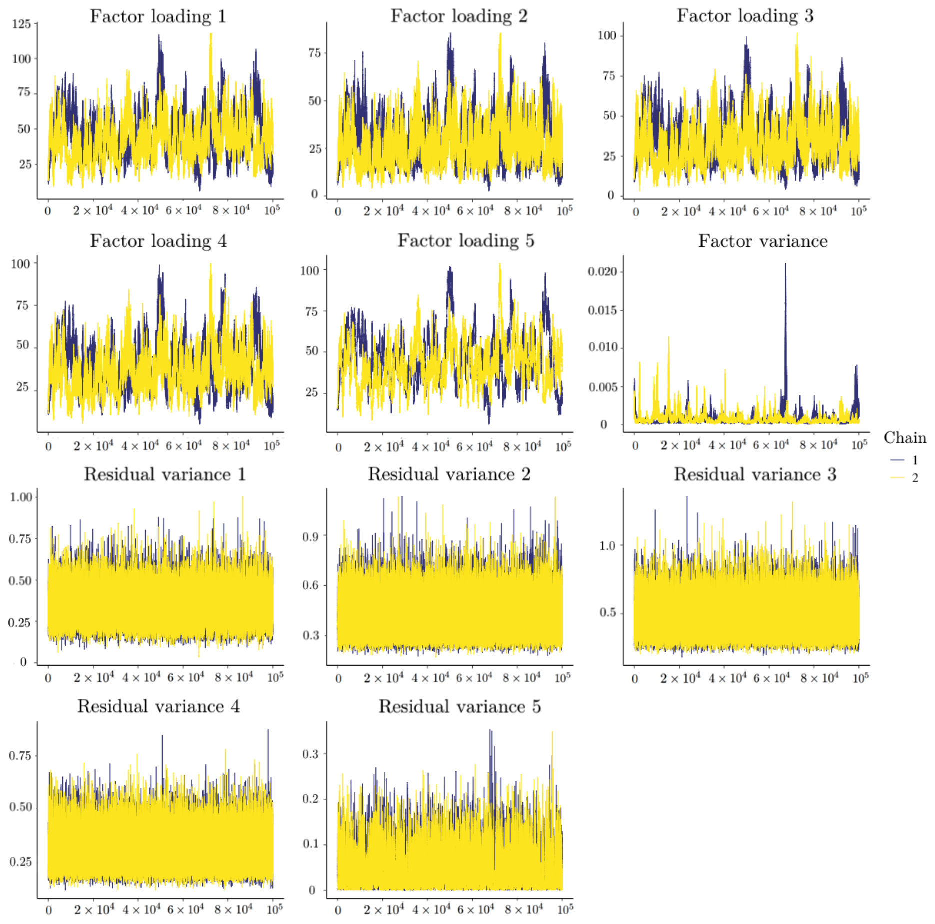


Figure 1: Trace Plots when $N = 50$ with a Positivity Constraint in Factor Analysis

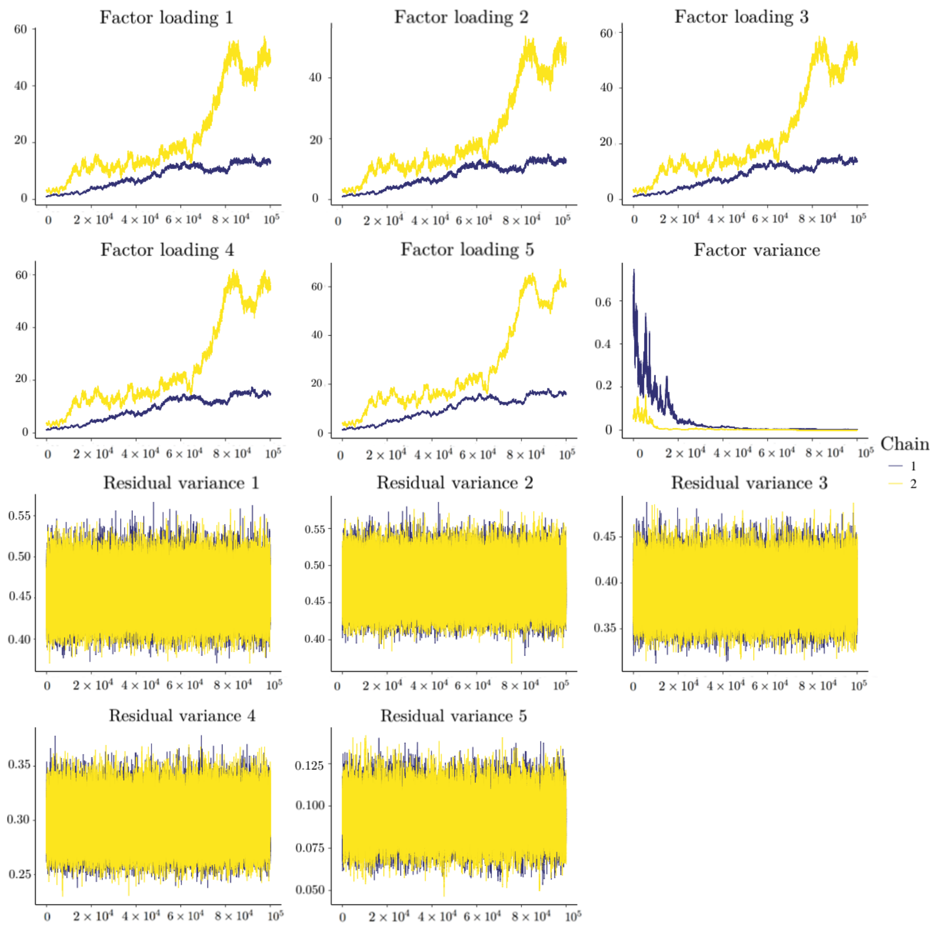


Figure 2: Trace Plots when $N = 1000$ with a Positivity Constraint in Factor Analysis

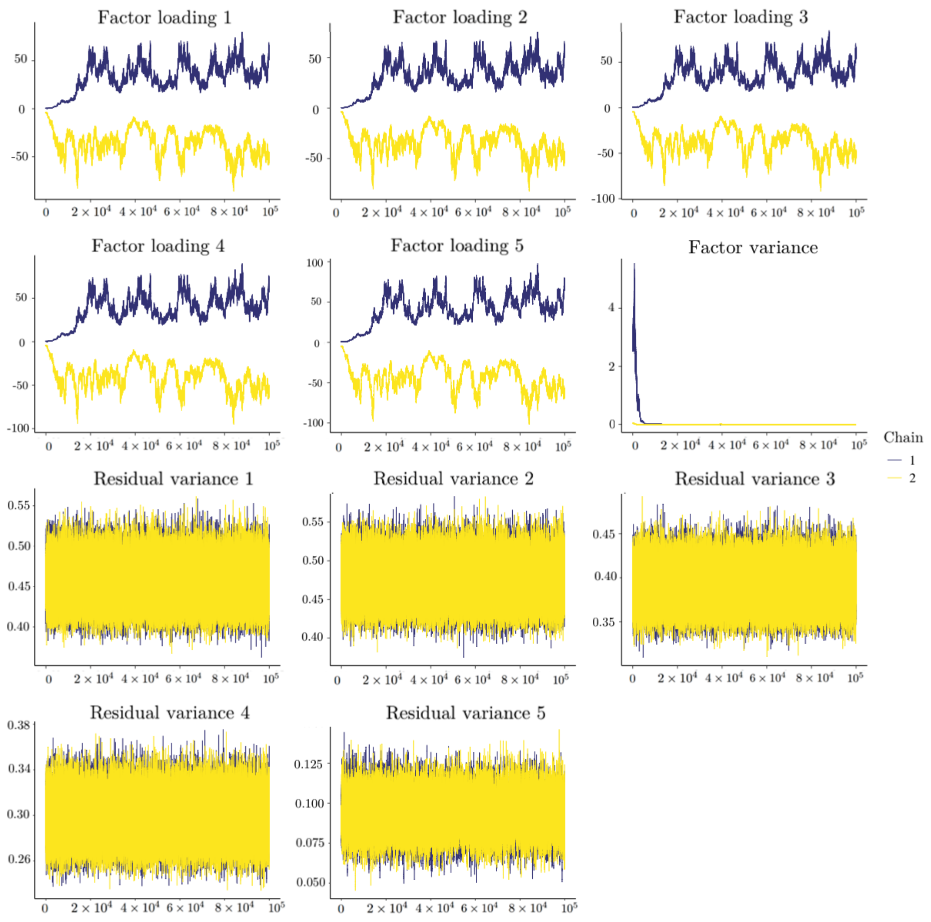


Figure 3: Trace Plots when $N = 1000$ without a Positivity Constraint in Factor Analysis

Table 7: Empirical Type II Error rates for $PSRF_{upper} > 1.1$, $PSRF > 1.1$, $MPSRF > 1.1$, and $|Geweke| > 1.96$ in the Factor Analysis Study without a Positivity Constraint

		$PSRF_{upper} > 1.1$						
N	T	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0	0	0.002	0.044	0.043	0.165	0.207
100		0	0	0.008	0.019	0.03	0.07	0.243
200		0	0.002	0.004	0.013	0.073	0.163	0.243
500		0	0	0	0.001	0.006	0.022	0.051
1000		0	0	0	0.001	0.006	0.047	0.059
		$PSRF > 1.1$						
N	T	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.004	0	0.005	0.142	0.12	0.241	0.302
100		0.006	0.002	0.038	0.05	0.11	0.182	0.371
200		0	0.006	0.025	0.031	0.187	0.342	0.436
500		0	0	0.003	0.004	0.015	0.056	0.104
1000		0	0	0.001	0.001	0.021	0.115	0.125
		$MPSRF > 1.1$						
N	T	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.003	0.001	0.011	0.268	0.271	0.431	0.438
100		0.009	0.002	0.05	0.112	0.206	0.478	0.54
200		0.001	0.007	0.037	0.069	0.347	0.645	0.728
500		0	0	0.003	0.006	0.031	0.109	0.21
1000		0	0	0.001	0.003	0.028	0.261	0.276
		$|Geweke| > 1.96$						
N	T	250	5×10^2	1.5×10^3	2.5×10^3	5×10^3	2.5×10^4	5×10^4
50		0.28	0.022	0.159	0.176	0.14	0.76	0.698
100		0.128	0.011	0.374	0.623	0.379	0.592	0.454
200		0.037	0.193	0.018	0.281	0.303	0.402	0.836
500		0.207	0.464	0.195	0.371	0.044	0.555	0.873
1000		0.041	0.088	0.228	0.012	0.101	0.378	0.135

gence assessment is critical to Markov chain Monte Carlo (MCMC) algorithms. Without appropriate convergence assessment, we cannot make reliable statistical inferences from the MCMC samples. Various quantitative diagnostic methods have been proposed for assessing convergence. The general recommendation is to use all the possible diagnostics because no method outperforms the others consistently. We endorse this recommendation if applying all diagnostic methods is feasible. However, there are situations where we cannot perform multiple convergence diagnostics. For example, in simulation studies, researchers rarely use multiple diagnostics to check convergence for all replications because it is time-consuming. Additionally, some software only provides one convergence diagnostic (e.g., BUGS and Mplus) and has created barriers for applied researchers to perform all convergence assessments. We do not object to the use of multiple or all convergence diagnostics simultaneously, but we would like to provide a guideline for applied researchers when resources are limited and it is not possible to perform multiple diagnostics. In the current paper, we focused on Gelman and Rubin's diagnostic (PSRF), Brooks and Gleman's multivariate diagnostic (MPSRF), and Geweke's diagnostic.

Previous studies that reviewed and/or compared different convergence diagnostics using hypothetical examples did not study their statistical properties such as Type I error rates and Type II error rates (Brooks and Roberts, 1998; Cowles and Carlin, 1996; El Adlouni et al., 2006). In this study, we evaluated these two statistical properties of the seven diagnostic criteria via simulation studies. Based on the results of simulation studies, we obtained a better understanding of the answers to the three unsolved questions listed in the introduction section. *For the first question, if we only can choose one diagnostic, which one should we used?* We recommend the upper bound of PSRF for three reasons. First, in terms of the Type I error rates, the upper bound of PSRF and PSRF required fewer iterations to achieve an acceptable Type I error rate ($\leq$ 5%), compared to MPSRF the Geweke's diagnostic (see Table 4). Second, in terms of the Type II error rates, PSRF led to higher Type II error rates than the upper bound of PSRF when the model was unidentified and the MCMC chains could not converge. $PSRF_{upper} > 1.1$ had the smallest Type II error rates among $PSRF_{upper} > 1.1$, $PSRF > 1.1$, $MPSRF > 1.1$, and $|Geweke| > 1.96$. Overall, balancing both the Type I error rate and Type II error rate, we recommend using the upper bound of PSRF. Third, PSRF and its upper bound could detect non-convergence due to bad mixing (e.g., the sign reflection invariance case) whereas Geweke's diagnostic could not. But we also need to note that PSRF is criticized for relying on over-dispersed starting values.

For the second question, when the number of estimated parameters is large, should we rely on the diagnostic per parameter (i.e., PSRF) or the multivariate diagnostic (i.e., MPSRF)? MPSRF yielded higher Type I and Type II error rates than PSRF and the upper bound of PSRF. Therefore, we still recommend the upper bound of PSRF over MPSRF.

For the third question, should we allow a small proportion of the parameters (e.g., 5%) to have significant convergence test results but still claim convergence *as a whole?* Comparing $PSRF_{upper,5\%} > 1.1$ and $PSRF_{upper} > 1.1$, the minimal number of iterations to control the analysis-wise Type I error rates below 5% did not differ dramatically. As the number of iterations increased, their Type I error rates were the same (0%). It is also difficult to define how small the proportion of the parameters should be in a widely acceptable way. In this paper, we used 5%, but one may would like to use 1% or 10%. Hence, we do not suggest allowing a small proportion of the parameters to have significant convergence diagnosis results but still claim convergence at the analysis level.

We echo the recommendation from previous studies, which advocate the use of all possible diagnostics when software and computational source are available. But when one has to choose one diagnostic, we recommend the upper bound of PSRF ($PSRF_{upper} > 1.1$). Even with a large number of parameters, we think it is better not to allow a 5% Type I error rate within each analysis. Additionally, we suggest using both quantitative diagnostics and visual inspection (e.g., trace plot) because trace plots provide extra information. For example, in simulation studies, one can randomly select several replications to check the trace plots, combined with the convergence rates from quantitative diagnostics.

Note

The simulation code is available at https://github.com/hduquant/Convergence-Diagno
stics.git. Correspondence should be addressed to Han Du, Pritzker Hall, 502 Portola Plaza, Los Angeles, CA 90095. Email: hdu@psych.ucla.edu.

References

Brooks, S. P. and Gelman, A. (1998). General methods for monitoring convergence of iterative simulations. *Journal of Computational and Graphical Statistics*, 7(4):434–455.

Brooks, S. P. and Roberts, G. O. (1998). Convergence assessment techniques for markov chain monte carlo. *Statistics and Computing*, 8(4):319–335.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*, 76(1).

Cowles, M. K. and Carlin, B. P. (1996). Markov chain monte carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, 91(434):883–904.

El Adlouni, S., Favre, A.-C., and Bobée, B. (2006). Comparison of methodologies to assess the convergence of markov chain monte carlo methods. *Computational Statistics & Data Analysis*, 50(10):2685–2701.

Erosheva, E. A. and Curtis, S. M. (2017). Dealing with reflection invariance in bayesian factor analysis. *Psychometrika*, 82(2):295–307.

Garren, S. T. and Smith, R. L. (2000). Estimating the second largest eigenvalue of a markov transition matrix. *Bernoulli*, pages 215–242.

Gelfand, A. E. and Smith, A. F. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, 85(410):398–409.

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2014). *Bayesian data analysis*, volume 2. Chapman & Hall, London.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, pages 457–472.

Geweke, J. (1992). Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments. In *Bayesian Statistics*, pages 169–193. University Press.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. (1996). Introducing markov chain monte carlo. In Gilks, W. R., Richardson, S., and Spiegelhalter, D. J., editors, *Markov chain Monte Carlo in practice*, pages 339 – 357. Chapman & Hall, London.

Hastings, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109.

Heidelberger, P. and Welch, P. D. (1983). Simulation run length control in the presence of an initial transient. *Operations Research*, 31(6):1109–1144.

Johnson, V. E. (1996). Studying convergence of markov chain monte carlo algorithms using coupled sample paths. *Journal of the American Statistical Association*, 91(433):154–166.

Lee, M. D. (2008). Three case studies in the bayesian analysis of cognitive models. *Psychonomic Bulletin & Review*, 15(1):1–15.

Liu, C., Liu, J., and Rubin, D. B. (1992). A variational control variable for assessing the convergence of the gibbs sampler. In *Proceedings of the American Statistical Association, Statistical Computing Section*, pages 74–78.

Lynch, S. M. (2007). *Introduction to applied Bayesian statistics and estimation for social scientists.* Springer Science & Business Media.

Marsman, M., Schönbrodt, F. D., Morey, R. D., Yao, Y., Gelman, A., and Wagenmakers, E.-J. (2017). A bayesian bird's eye view of 'Replications of important results in social psychology'. *Royal Society open science*, 4(1):160426.

Muthén, L. and Muthén, B. (1998–2017). *Mplus User's Guide. 8th edition.* Los Angeles, CA: Author.

Plummer, M., Best, N., Cowles, K., and Vines, K. (2015). Package 'coda'. *URL http://cran. r-project. org/web/packages/coda/coda. pdf, accessed January*, 25:2015.

Raftery, A. and Lewis, S. (1991). How many iterations in the gibbs sampler?

Spiegelhalter, D. J., Thomas, A., Best, N. G., Gilks, W., and Lunn, D. (1996). Bugs: Bayesian inference using gibbs sampling. *Version 0.5,(version ii) http://www. mrc-bsu. cam. ac. uk/bugs*, 19.

Van de Schoot, R., Kaplan, D., Denissen, J., Asendorpf, J. B., Neyer, F. J., and Van Aken, M. A. (2014). A gentle introduction to bayesian analysis: Applications to developmental research. *Child Development*, 85(3):842–860.

Van de Schoot, R., Winter, S. D., Ryan, O., Zondervan-Zwijnenburg, M., and Depaoli, S. (2017). A systematic review of "bayesian" articles in psychology: The last 25 years. *Psychological Methods*, 22(2):217.

Walker, L. J., Gustafson, P., and Frimer, J. A. (2007). The application of bayesian analysis to issues in developmental research. *International Journal of Behavioral Development*, 31(4):366–373.

Journal of Behavioral Data Science, 2022, *2* (2), 73–98.
DOI: `https://doi.org/10.35566/jbds/v2n2/suzuki`

Relative Predictive Performance of Treatments of Ordinal Outcome Variables across Machine Learning Algorithms and Class Distributions

Honoka Suzuki and Oscar Gonzalez

University of North Carolina at Chapel Hill, Chapel Hill, NC, USA
hsuzuki@unc.edu

Abstract. Ordinal variables, such as those measured on a five-point Likert scale, are ubiquitous in the behavioral sciences. However, machine learning methods for modeling ordinal outcome variables (i.e., ordinal classification) are not as well-developed or widely utilized, compared to classification and regression methods for modeling nominal and continuous outcomes, respectively. Consequently, ordinal outcomes are often treated "naively" as nominal or continuous outcomes in practice. This study builds upon previous literature that has examined the predictive performance of such naïve approaches of treating ordinal outcome variables compared to ordinal classification methods in machine learning. We conducted a Monte Carlo simulation study to systematically assess the relative predictive performance of an ordinal classification approach proposed by Frank and Hall (2001) against naïve approaches according to two key factors that have received limited attention in previous literature: (1) the machine learning algorithm being used to implement the approaches and (2) the class distribution of the ordinal outcome variable. The consideration of these important, practical factors expands our knowledge on the consequences of naïve treatments of ordinal outcomes, which are shown in this study to vary substantially according to these factors. Given the ubiquity of ordinal measures coupled with the growing presence of machine learning applications in the behavioral sciences, these are important considerations for building high-performing predictive models in the field.

Keywords: Ordinal classification · Machine learning · Predictive performance · Class imbalance · Measurement scale

1 Introduction

In supervised learning, ordinal classification or equivalently ordinal regression, refers to a classification task where classes of the categorical outcome variable have an inherent ordering. This distinguishes it from nominal multi-class classification, where the classes are unordered. Ordinal classification is also distinct from regression where the outcome variable is continuous, because the numeric labels of the ordinal classes do not indicate equal spacing between adjacent classes. Ordinal measures are ubiquitous in a variety of disciplines, including the behavioral sciences. For example, human response data are often captured in Likert-type scales, such as those with response levels ranging from *strongly disagree* to *strongly agree* or *poor* to *excellent*.

Although not as well-developed and well-studied compared to nominal classification and regression (Ben-David, Sterling, & Tran, 2009; Gutierrez, Perez-Ortiz, Sanchez-Monedero, Fernandez-Navarro, & Hervas-Martinez, 2016), machine learning methods for ordinal classification have been developed by many researchers. Some of these methods are modified versions of specific algorithms developed to handle ordinal outcome variables. This has been particularly popular with support vector machines (Chu & Keerthi, 2007; Crammer & Singer, 2005; Gu, Sheng, Tay, Romano, & Li, 2015; Herbrich, Graepel, & Obermayer, 2000) and neutral networks (Cheng, Wang, & Pollastri, 2008; Deng, Zheng, Lian, Chen, & Wang, 2010; Fernandez-Navarro, Riccardi, & Carloni, 2014). Rather than modifying specific algorithms, researchers have also developed ordinal classification methods that can be implemented with multiple algorithms (Cardoso & da Costa, 2007; Frank & Hall, 2001; Lin & Li, 2012). For example, Frank and Hall (2001) proposed an approach to decompose an ordinal classification task into multiple binary classification tasks while retaining the ordinal information among classes, where any algorithm can be used as the base binary classifier, making it an algorithm-independent approach.

However, in practice, without strong familiarity with ordinal classification methods, machine learning researchers and practitioners may choose to implement a more "naïve" and easier approach to modeling ordinal outcome variables (Gutierrez et al., 2016). This involves casting the ordinal outcome variable as a nominal variable, in which case the task at hand reduces to nominal multi-class classification. Similarly, the ordinal outcome variable may be cast as a continuous variable, in which case the task at hand becomes regression. To obtain integer-valued predictions in this case, some form of post-processing of the real-valued predictions, such as rounding, may be required (Kramer, Widmer, Pfahringer, & de Groeve, 2000). We refer to these naïve treatments of ordinal outcome variables as naïve classification and naïve regression, respectively.

Despite the developments in the ordinal classification literature, there are several possible reasons why researchers and practitioners may choose to implement a naïve approach. Bürkner and Vuorre (2019) discuss how it is common practice to treat ordinal measures as if continuous in the context of traditional statistical methods (e.g., t-tests, ANOVA, ordinary least squares regression). They provide several reasons for this, including hesitation due to perceived complexity in implementation or interpretation of ordinal approaches, difficulty in deciding which ordinal model to choose, or skepticism from journal editors and reviewers for using a "non-standard" approach (Bürkner & Vuorre, 2019). Although their discussion was in the context of traditional statistical methods, these reasons conceivably apply to machine learning contexts as well. Other factors may also include unfamiliarity or unavailability of accessible software to readily implement ordinal classification methods.

Given that naïve treatments of ordinal outcome variables are not uncommon, it is important to understand the consequences of implementing such naïve approaches. In the statistical literature, the consequences of treating ordinal measures as if continuous, rather than nominal, has specifically received attention. Liddell and Kruschke (2018) found this practice to be particularly common in psychological research, after surveying articles published in several highly ranked psychology journals. Motivated by this finding, the authors showed how systematic errors in analyses can arise from treating ordinal measures as continuous, including inflated Type I and Type II errors and misleading effect size estimates, and they suggest using models that allow for a proper treatment of ordinal variables (Liddell & Kruschke, 2018) . In the machine learning literature, the analogous investigation of the consequences of using naïve approaches are perhaps studies that compare ordinal classification methods against naïve classification or naïve regression in terms of predictive performance (Ben-David et al., 2009; Cardoso & da Costa, 2007; Chu & Keerthi, 2007; Frank & Hall, 2001; Herbrich et al., 2000; Kramer et al., 2000). In such studies, the naïve approaches are typically, although not always, shown to have lower predictive performance compared to an ordinal classification method, given that naïve approaches give a less than ideal treatment of ordinal variables by discarding the ordinal information (naïve classification) or assuming equal spacing between adjacent classes (naïve regression).

The present study builds upon this literature by investigating how ordinal classification methods perform relative to naïve classification and naïve regression according to key factors, including the machine learning algorithm being used to implement the approaches, the number of classes, and the degree of class imbalance present in the ordinal outcome variable. We do so by conducting a Monte Carlo simulation study to systematically evaluate predictive performance across different treatments of ordinal outcome variables, each implemented with multiple machine learning algorithms and crossing different levels of class imbalance with different numbers of classes in the ordinal outcome variable. This investigation is related to, but extends in important ways, previous studies in this line of research. For example, Gutierrez et al. (2016) examined the performance of naïve approaches and various ordinal classification methods on several datasets with varying numbers of ordinal classes. However, naïve approaches were only implemented using support vector machines, and varying degrees of class imbalance in the ordinal outcome variable were not varied or examined systematically with the number of classes. Cardoso and da Costa (2007) examined the performance of their data reduction method for ordinal classification compared to naïve approaches, implemented with support vector machines and neural networks. However, their study also did not systematically examine the impact of class distributions in the ordinal outcome variable. Similarly, Ben-David et al. (2009) used logistic regression and support vector machines to compare the performance of ordinal classification methods based on these algorithms compared to naïve classification. However, they did not examine class distributions or the performance of naïve regression using these same algorithms.

Researchers and practitioners routinely work with multiple algorithms in a given task, and many ordinal variables in real data tend to exhibit class imbalance (Baccianella, Esuli, & Sebastiani, 2009). As such, investigating how predictive performance compares across different treatments of the ordinal outcome variables in light of these practical considerations would expand our knowledge on the consequences of resorting to a naïve approach under various conditions and encourage more informed choices around the treatment of ordinal variables in practice.

2 Methods

We compared the predictive performance of three treatments of ordinal outcome variables: as a nominal variable in naïve classification, as a continuous variable in naïve regression, and as an ordinal variable in an ordinal classification method. For the ordinal classification method, we chose to employ the algorithm-independent approach proposed by Frank and Hall (2001). We chose this approach because it can be implemented with multiple algorithms, which allows for more intuitive comparisons in the relative predictive performance of the ordinal classification method across algorithms. Moreover, the approach is simple and intuitive. Given the practical barriers to employing ordinal classification methods discussed above, it may be most useful to examine the relative performance of an ordinal classification method that practitioners are most realistically likely to implement and that does not require much heavy lifting from naïve approaches.

2.1 Frank and Hall (2001) Approach

Given an ordinal classification task with k ordinal classes, the Frank and Hall (2001) approach ("FH approach" hereafter) involves training $k-1$ binary classifiers on $k-1$ modified copies of the original dataset. The j^{th} binary classifier is trained on a modified outcome variable that is a binary indicator for whether or not the original outcome is greater than the j^{th} ordered class. The predictors remain unchanged. For example, with $k = 3$ (class 1 < class 2 < class 3), two binary classifiers are trained, where the first classifier predicts whether or not the outcome is greater than class 1 (i.e., class 2 or 3), and the second classifier predicts whether or not the outcome is greater than class 2 (i.e., class 3). Using the predicted probabilities from the $k-1$ binary classifiers, the predicted probability that an observation belongs to each of the k classes is obtained in the following manner:

$$\begin{gathered}
P\left(Y = \mathit{Class}\ 1|X\right) = 1 - P(Y > \mathit{Class}\ 1|X) \\
P\left(Y = \mathit{Class}\ k|\,X\right) = P\left(Y > \mathit{Class}\ k|\,X\right) \\
P\left(Y = \mathit{Class}\ j|\,X\right) = P\left(Y > \mathit{Class}\ j-1|\,X\right) - P\left(Y > \mathit{Class}\ j|\,X\right), \\
j = 2, \ldots, k-1
\end{gathered}$$

Once the k predicted probabilities are calculated per observation, an observation is assigned to the class with the greatest predicted probability.

In their study, Frank and Hall (2001) demonstrated this approach on many benchmark regression datasets by discretizing the continuous outcomes into k balanced ordinal classes. They used C4.5 as the base algorithm and evaluated predictive performance with accuracy (1 minus misclassification rate). Using k = 3, 5, and 10, they found higher accuracy associated with the FH approach across most datasets, compared to naïve classification (i.e., C4.5 treating the outcome as a nominal variable). They also found this performance gap between the FH approach and naïve classification to increase with k. The present study extends these evaluations by implementing the FH approach with more algorithms besides C4.5, considering imbalanced ordinal classes, comparing the predictive performance of the FH approach against naïve regression in addition to naïve classification, and considering additional performance metrics besides the misclassification rate.

2.2 Algorithms Used

We implemented each of the three approaches (naïve classification, naïve regression, and FH approach) with two machine learning algorithms: classification and regression trees (CART; Breiman, Friedman, Olshen, & Stone, 2017) and random forests (Breiman, 2001). These tree-based algorithms have become increasingly popular among researchers in many disciplines including psychology, due to their desirable qualities such as ease of application and interpretability (Strobl, Malley, & Tutz, 2009). Despite their popularity, these algorithms' performance across naïve and ordinal classification methods has not been examined as extensively compared to other algorithms, such as support vector machines and neural networks.

2.3 Performance Metrics

Although the misclassification rate is a popular performance metric for classification tasks, it is not ideal for ordinal classification tasks because it gives equal penalty to all types of misclassifications (Gaudette & Japkowicz, 2009). For example, misclassifying a *strongly disagree* response as *disagree* is not as detrimental of an error as misclassifying a *strongly disagree* response as *strongly agree*. As such, we used two additional performance metrics besides the misclassification rate to evaluate the approaches: mean absolute error (MAE) and Spearman's correlation coefficient. With observed numeric class labels y and predicted numeric class labels $\hat{y}$, mean absolute error is calculated as $N^{-1}\sum_{i=1}^{N}|y_i - \hat{y}_i|$. Spearman's correlation coefficient is calculated as $1 - 6\sum_{i=1}^{N} {D_i}^2/[N(N^2-1)]$, where D_i refers to the difference between the rank order of y_i and that of $\hat{y}_i$. These measures better account for the severity of error based on the ordinal information by using the numeric labels of the ordinal classes. These metrics have also been used in several studies that have examined the performance of ordinal classification approaches (Cardoso & da Costa, 2007; Gutierrez et al., 2016; Kramer et al., 2000).

In addition to evaluating overall predictive performance of each approach, we also evaluated the predictive performance of each approach at the class-level. For class-level performance, we examined the F1 score per class. The F1 score captures a balance of precision (proportion of true positives out of predicted positives) and recall (proportion of true positives out of actual positives) and can be calculated as $2{*}Precision * Recall/(Precision + Recall)$. A "positive" case in this context refers to an observation belonging to a given class, and a "negative" case refers to an observation belonging to all other classes.

2.4 Simulation Design and Analysis Plan

To investigate our research question, we conducted Monte Carlo simulations using the R statistical software (R Core Team, 2022). There were two simulation factors: the number of ordinal classes in the outcome variable (k = 3, 5, 7) and the degree of class imbalance present in the ordinal outcome variable (we termed them *balanced*, *slightly imbalanced*, *imbalanced* class distributions). This gave a total of nine conditions, and each condition contained 500 replications. For each replication, we simulated a dataset of 2,000 observations using the *mlbench.friedman1* function from the *mlbench* package (Leisch & Dimitriadou, 2021). This generates a benchmark regression dataset with ten predictors (x_1 through x_{10}) from a uniform distribution bounded by 0 and 1 and an error term from the standard normal distribution. Then, a subset of these predictors is used to generate a continuous outcome variable y, where

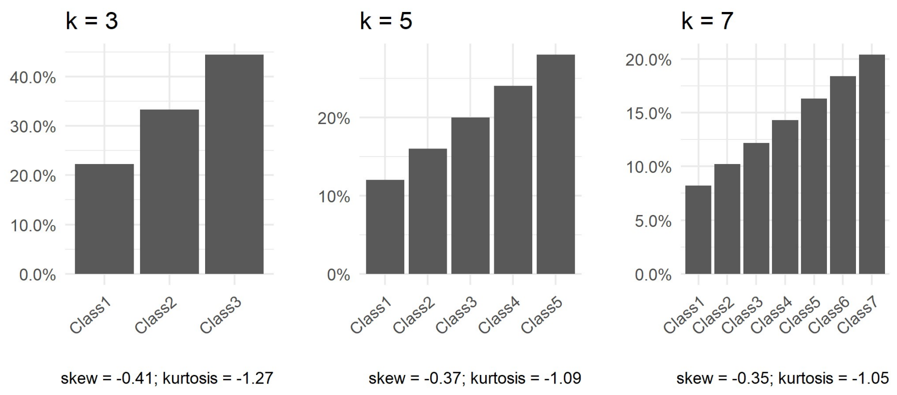
$$y = 10\sin(x_1x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + e$$

which we discretized into k balanced, slightly imbalanced, or imbalanced classes, depending on the condition. For the balanced condition, each class contained $1/k$ of the observations. For the slightly imbalanced condition, the j^{th} class contained $1/k + [j - 0.5(k + 1)]/k^2$ of the observations. For the imbalanced condition, the j^{th} class contained

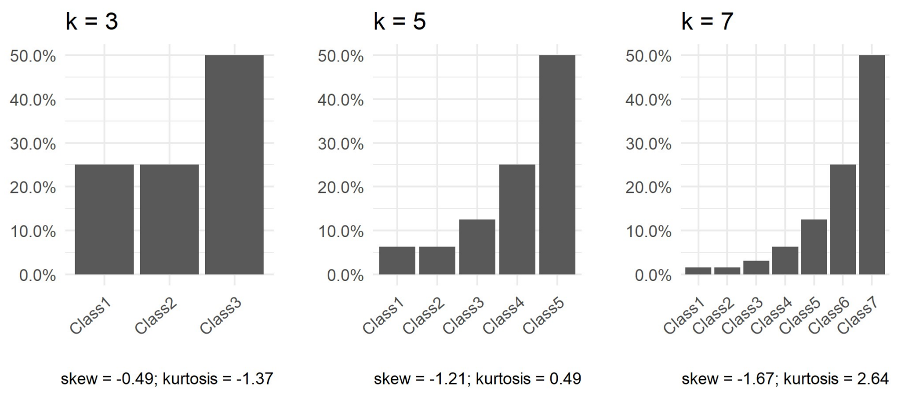
$$\begin{gathered} 2^{-(k-1)},\, j = 1 \\ 2^{-(k-j+1)},\, j = 2, \ldots, k \end{gathered}$$

of the observations. To better visualize these proportions, Figure 1 presents the class distributions that result from the above rules for k = 3, 5, and 7, along with measures of skewness and kurtosis for each distribution. Note that these rules can be used to generate imbalanced and slightly imbalanced class distributions for any value of k.

In each dataset, we implemented the two algorithms, CART and random forest, using naive classification, naïve regression, and the FH approach, for a total of six models per replication. Models were built using the *caret* package (Kuhn, 2022), by calling the *rpart* (Therneau & Atkinson, 2022) and *rf* (Liaw & Wiener, 2002) methods for CART and random forest, respectively. We trained these models and tuned hyperparameters with five-fold cross validation using



(a) Slightly imbalanced condition



(b) Imbalanced condition

Figure 1: Class distributions for the slightly imbalanced (Panel a) and imbalanced (Panel b) conditions

50% (N = 1,000) of the dataset and evaluated their predictive performance on the remaining 50% (N = 1,000) of the dataset. Note that for naïve regression, real-valued predictions were rounded to the nearest class. For CART, we tuned the complexity parameter, which is the factor by which any additional split attempted in the tree must decrease the prediction error. For random forests, we tuned the number of predictors that are randomly selected as split candidates at each split in a tree. For each of the six models per replication, we recorded the misclassification rate, MAE, and Spearman's correlation as measures of overall predictive performance, and we recorded the F1 score for each of k classes as a measure of class-level performance.

Within each algorithm in each condition, we examined the mean difference in overall predictive performance (for each of the three overall performance metrics) between the FH approach and naïve classification and between the FH approach and naïve regression. We first qualitatively examined the patterns by plotting the distributions of the performance metrics via boxplots. We then conducted paired samples t-tests and calculated effect sizes of the paired differences in performance across the approaches for each algorithm and condition to quantitatively examine patterns. We similarly calculated effect sizes for the paired differences in class-level performance between the FH approach and naïve approaches for each algorithm and condition.

In addition to the simulation study, we demonstrate the practical implementation of the FH approach in Appendix A, which contains step-by-step R code to carry out the FH approach using CART and random forests (but can easily be adapted to use any other algorithm) with an empirical example, as well as the resulting overall and class-level performance of the FH and naïve approaches on this empirical dataset.

3 Results

The overall performance results using Spearman's correlation and MAE led to largely the same conclusions. For brevity, we focus our discussion on the results using Spearman's correlation. There were a few differences in results when evaluating overall performance using the misclassification rate, which we summarize towards the end of this section. Results based on MAE and the misclassification rate can be found in Appendix B.

3.1 CART Implementation

The CART implementation produced results that extend Frank and Hall's (2001) findings well. Figure 2 presents the distribution of the six models' predictive performance in terms of Spearman's correlation across the 500 replications. Each square subplot represents a condition defined by the number of classes k and the degree of class imbalance. Qualitatively, Figure 2 shows that with CART, the FH approach outperformed (i.e., higher Spearman's correlations) both naïve classification and naïve regression across all nine conditions, including the imbalanced and slightly imbalanced conditions. Further, the performance gaps between the FH and naïve approaches generally appear to grow with a larger k within each level of class imbalance.

To examine these observations quantitatively, we conducted paired samples t-tests within each set of 500 replications (i.e., within each of the nine conditions) per algorithm to compare the overall performance of the FH approach to each of the naive approaches. The mean paired difference (and standard deviation of the difference) in Spearman's correlation for each algorithm in each condition are presented in Table 1, as well as the effect sizes in Cohen's d for those differences (Cohen, 1988). Differences were calculated as Spearman's correlation of the FH approach minus Spearman's correlation of the naïve approaches. Thus, a positive mean difference and effect size indicate the FH approach performing better, and a negative mean difference and effect size indicate the FH approach performing worse compared to the naïve approaches. Effect sizes are also visualized in Figure 3.

Table 1: Mean paired differences (and standard deviations) in overall predictive performance in terms of Spearman's correlation between the FH approach and naive approaches for each algorithm in each simulation condition; effect sizes of the paired differences

k	Degree of class balance	CART	
		Naïve Classification	Naïve Regression
3	Balanced	.015 (.023)*; **.674**	.035 (.025)*; **1.388**
	Slightly Imbalanced	.021 (.024)*; **.850**	.037 (.027)*; **1.384**
	Imbalanced	.023 (.028)*; **.797**	.038 (.027)*; **1.425**
5	Balanced	.047 (.023)*; **2.079**	.061 (.022)*; **2.788**
	Slightly Imbalanced	.049 (.022)*; **2.209**	.048 (.019)*; **2.534**
	Imbalanced	.033 (.028)*; **1.170**	.058 (.025)*; **2.267**
7	Balanced	.066 (.026)*; **2.527**	.059 (.015)*; **3.836**
	Slightly Imbalanced	.069 (.026)*; **2.681**	.063 (.018)*; **3.547**
	Imbalanced	.036 (.029)*; **1.212**	.065 (.027)*; **2.386**
		Random Forest	
		Naïve Classification	Naïve Regression
3	Balanced	-.003 (.010)*; -.315	-.001 (.012); -.068
	Slightly Imbalanced	-.002 (.010)*; -.179	.003 (.011)*; .297
	Imbalanced	.002 (.011)*; .218	.006 (.014)*; .426
5	Balanced	-.001 (.008)*; -.167	-.027 (.008)*; **-3.234**
	Slightly Imbalanced	.000 (.008); -.014	-.023 (.008)*; **-2.737**
	Imbalanced	-.002 (.013); -.127	-.018 (.015)*; **-1.211**
7	Balanced	-.001 (.008); -.082	-.038 (.007)*; **-5.149**
	Slightly Imbalanced	.001 (.009); .130	-.036 (.008)*; **-4.343**
	Imbalanced	-.003 (.014)*; -.178	-.024 (.017)*; **-1.471**

Note. $^*p < \frac{.05}{36}$ (Bonferroni-corrected); **bold** indicates moderate or large effect size ($|d| > 0.5$). Positive mean differences and effect sizes indicate the FH approach performing better than the naïve approach.

Table 1 shows that for the CART implementation, as qualitatively observed in Figure 2, the FH approach had significantly better performance than both naïve approaches. This is indicated by the mean differences in predictive performance being positive and significant ($p < \frac{.05}{36}$; Bonferroni-corrected alpha level for multiple testing) for both naïve classification and naïve regression in all nine conditions. Further, effect sizes were all at least in the moderate range ($d > .5$), and effect sizes grew more positive with a larger k, as illustrated in Figure 3. Overall, we observed comparable effect sizes between balanced and slightly imbalanced classes, while imbalanced classes tended to have smaller effect sizes. These results indicate that with CART, the FH approach resulted in improved predictive performance over both naïve approaches, and that this performance boost was most prevalent in conditions with more classes and more class balance.

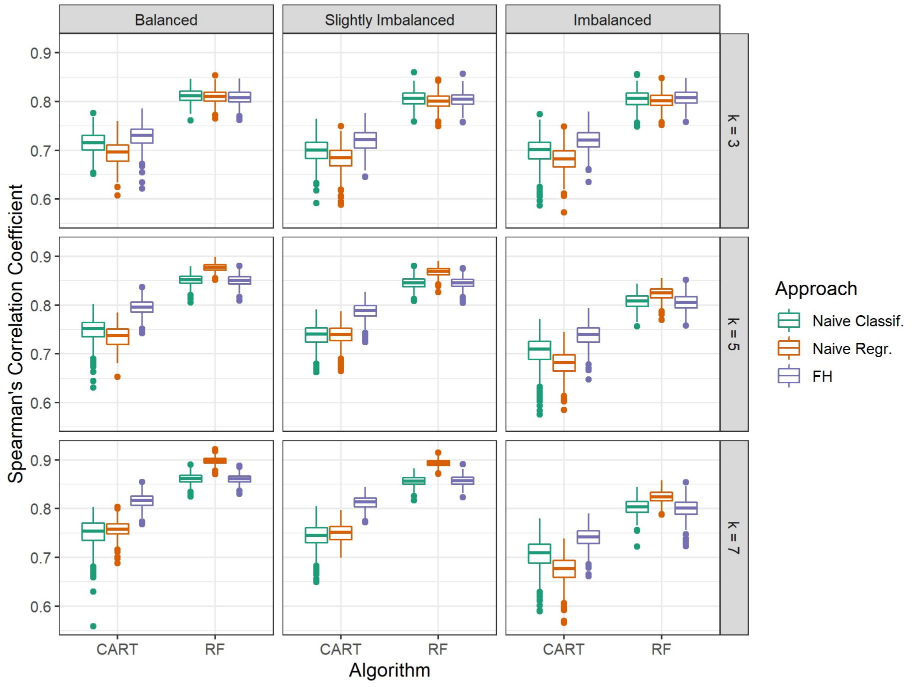


Figure 2: Distribution of overall predictive performance (Spearman's correlation) of the six models across replications in each simulation condition

Figure 4(a) plots the effect sizes of the paired differences in class-level F1 scores between the FH and naïve approaches for the CART implementation in each condition. It is interesting to note that in all conditions, not all class-level effect sizes were positive and at least moderate ($d > .5$), meaning class-level performance was only higher for the FH approach in some classes, even though overall performance was higher. For example, there appeared to be a pattern in nearly all conditions, where there was a negligible difference in performance between the FH approach and naïve classification for the "end" classes (i.e., classes 1 and k), as indicated by near-zero effect sizes, but the FH approach outperformed naïve classification for the "middle" classes (i.e., classes $2, \ldots, k-1$), as indicated by positive effect sizes. There were no such apparent patterns for naïve regression, but there were similarly some classes where the FH approach outperformed naïve regression and other classes where there were negligible differences. These class-level effect sizes indicate that the improvement in overall predictive performance associated with the FH approach in a given condition did not necessarily come from a uniform improvement in performance across *all* classes, but only in some.

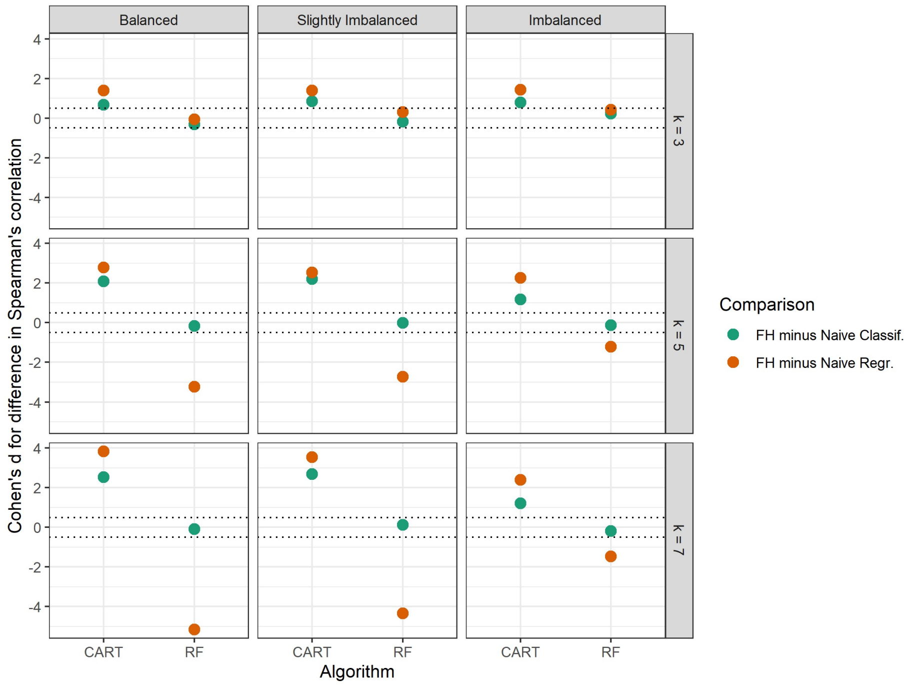


Figure 3: Effect sizes for the paired difference in overall predictive performance (Spearman's correlation) between the FH approach and naïve approaches for each algorithm in each simulation condition. Dashed lines appear at $|d| = 0.5$.

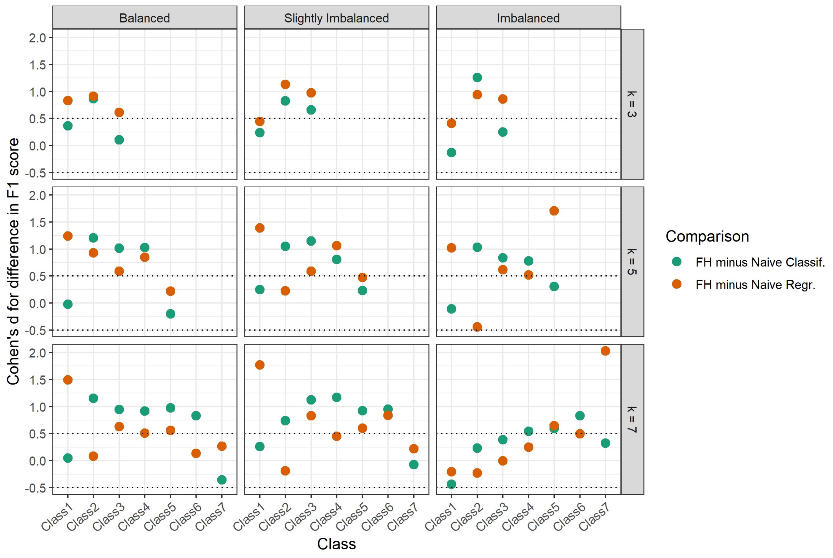
3.2 Random Forest Implementation

The random forest implementation produced less intuitive results compared to the CART implementation. First, Figure 2 shows that when comparing the FH approach to naïve classification, there were virtually no differences in overall predictive performance between these two approaches in all conditions, meaning that the FH approach did not perform any better than naïve classification. Second, Figure 2 shows that when comparing the FH approach to naïve regression, in conditions with $k = 3$, these two approaches also performed similarly, meaning the FH approach did not perform any better than naïve regression, either. However, in conditions with a larger k, naïve regression outperformed the FH approach. The mean differences and effect sizes in Table 1 corroborate these observations. While we did observe some significant mean differences ($p < \frac{.05}{36}$) in certain conditions for naïve classification, all effect sizes were small ($|d| < .5$). For naïve regression, we observed negative and significant mean differences with large effect sizes in conditions with $k = 5$ and 7. Figure 3 also illustrates these patterns. The green (naïve classification) effect sizes hovered around zero in all conditions, indicating no difference in overall performance between the FH approach and naïve classification. The orange (naïve regression) effect sizes, under $k = 3$, also hovered around zero, but with a larger k, they were negative. These effect sizes grew more negative with more class balance, indicating that naïve regression increasingly outperformed the FH approach with more class balance.

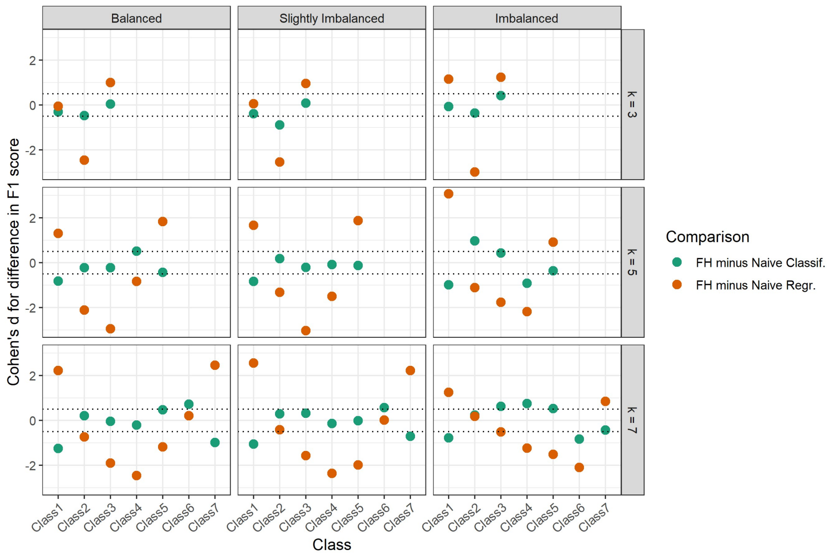
Figure 4(b) plots the effect sizes for the paired differences in class-level F1 scores between the FH approach and naïve approaches for the random forest implementation in each condition. For naïve classification, effect sizes generally hovered around zero in most conditions, indicating no difference in class-level performance between the FH approach and naïve classification. For naïve regression, there appeared to be a pattern in nearly all conditions where the "end" classes (i.e., classes 1 and k) had positive effect sizes, indicating that the FH approach outperformed naïve regression, but the "middle" classes (i.e., classes $2, \ldots, k-1$) had negative effect sizes, indicating that the FH approach performed worse than naïve regression. These class-level effect sizes show that the relative class-level performance of these approaches can differ according to class and may not all be in the same direction as the overall relative performance.

3.3 Summary of Results

With the CART implementation, the FH approach had significantly better overall performance than both naïve classification and naïve regression across all nine conditions, including those with imbalanced and slightly imbalanced class distributions. The overall performance gap between the FH approach and naïve approaches grew with k. With the random forest implementation, the FH approach did not perform differently from naïve classification in any meaningful way across all nine conditions. However, the FH approach performed significantly worse compared to naïve regression in conditions with a larger k, and the performance gap increased with more class balance. In both CART and random forest



(a) CART



(b) Random forest

Figure 4: Effect sizes for the paired difference in class-level predictive performance (F1 score) between the FH approach and naïve approaches for CART (Panel a) and random forests (Panel b) in each simulation condition. Dashed lines appear at $|d| = 0.5$.

implementations, we found that in a given condition, class-level performance was not uniformly better or worse for the FH approach across classes according to the overall performance results. For example, even when overall performance was substantially higher for the FH approach than a naïve approach in a given condition, this did not mean that the FH approach had accordingly higher class-level performance in each class for that condition.

Using the misclassification rate as the overall performance metric led to largely the same findings as Spearman's correlation throughout the CART and random forest implementations. The exception was that in the random forest implementation, the FH approach had comparable overall performance to both naïve classification and naïve regression across all conditions, rather than naïve regression outperforming the FH approach in some conditions. Given that the misclassification rate is an unsuitable performance metric for ordinal classification tasks, we do not expand on these findings. However, this does reveal that different conclusions can be made from using different overall performance metrics, highlighting the importance of using a metric that is most suited to the task at hand.

4 Discussion

Ordinal measures are ubiquitous, but given limitations in familiarity or availability of machine learning methods for ordinal classification, they may not always be treated as an ordinal variable in practice. In this study, we aimed to expand our understanding of the impacts of treatments of ordinal outcome variables on predictive performance across various conditions. Specifically, we used Monte Carlo simulations to examine the relative predictive performance of an ordinal classification method, namely the FH approach, against naïve classification and naïve regression according to the machine learning algorithm being implemented, the number of classes in the ordinal outcome variable, and the degree of class imbalance in the ordinal outcome variable.

Our results differed substantially across algorithms. With the CART implementation, results aligned well with and extended Frank and Hall's (2001) findings. The FH approach was associated with a higher overall predictive performance compared to both naïve classification and naïve regression, and this pattern held across all conditions, even in the presence of class imbalance. The overall performance gap increased with the number of classes and was largest among balanced classes, indicating that the benefit of treating the outcome as an ordinal variable by implementing the FH approach is greatest when there are many, balanced classes.

On the other hand, we found some divergent results with the random forest implementation. The FH approach had comparable overall performance to naive classification in all conditions, and the FH approach performed worse than naive regression in conditions with more classes and more class balance. One possible explanation for this could be tied to the fact that random forests are ensemble learners that, in general, tend to have better predictive performance than weak learners like CART and C4.5. Thus, it is possible that the naïve approaches implemented with random forests already provided decent predictive performance that there may not have been as much to be gained from the implementation of the FH approach. Further, with a larger number of classes, the ordinal outcome variable might become better approximated as a continuous outcome, and perhaps that is why we observed naïve regression to perform particularly well in those conditions. In a similar study that examined the performance of various algorithm-independent ordinal classification methods, Hühn and Hüllermeier (2008) theorized that algorithms with more complex and flexible decision boundaries benefit less from incorporating the ordinal information among the classes of the outcome variable. This is consistent with findings from our study, as the random forest implementation of the FH approach, treating the outcome as an ordinal variable, did not result in increased predictive performance over the naïve approaches, whereas the CART implementation did.

In sum, these findings illustrate that the relative predictive performance of the different treatments of ordinal outcome variables varies across algorithms and conditions. In other words, the gain in overall predictive performance from treating an ordinal outcome properly as an ordinal variable by implementing the FH approach can depend on the number of classes, the degree of class imbalance present, and the algorithm being used. There is not always an improvement in overall predictive performance associated with the FH approach, and sometimes, naïve approaches may perform better than the FH approach. As such, there is no one approach that is always best, suggesting a need for careful and deliberate choices in the treatment of ordinal outcomes to achieve optimal predictive performance in machine learning.

4.1 Limitations and Future Directions

There are several limitations associated with this study. First, the ordinal outcome variable in our simulated datasets were not "real" ordinal data, as the outcome variable was originally a continuous variable which was discretized. As such, the ordinal class structure may have been artificially accentuated compared to what is conceivable in naturally occurring ordinal data (Hühn & Hüllermeier, 2008). We used such simulated outcomes in order to be able to systematically manipulate and examine the influence of the number of classes and the degree of class imbalance in the ordinal outcome variable, which was a main goal of the study. Second, our findings are limited to the two specific algorithms, CART and random forest, that we implemented in this study. We chose these two algorithms as they have not received as much attention in the ordinal classification literature. However, there are many other binary classifiers that can be implemented with the FH approach, as this is an algorithm-independent approach. Given the surprising results we found with the random forest implementation, it would be interesting to study whether the same holds for other related methods, such as boosted trees (Hastie, Tibshirani, & Friedman, 2009). Similarly, we only examined one ordinal classification method to compare against naïve classification and naïve regression. Future studies should examine how other algorithm-independent ordinal classification methods provide similar or divergent results. Another direction for future study is to examine different shapes of class distributions. In this study, we only examined three levels of class imbalance, and the skewness of the imbalanced and slightly imbalanced distributions were in the same direction (i.e., all upwards sloping, where the most frequent class was class k). It would thus be interesting to examine how results may change with different shapes (e.g., downwards sloping with class 1 being the most frequent class, or a random pattern where class frequencies do not increase or decrease uniformly with class labels) and how this may impact class-level performance. Lastly, while Spearman's correlation and MAE provide more suitable overall performance metrics for ordinal classification tasks than the misclassification rate, they are not the only metrics available, nor are they free of flaws themselves. A major limitation of MAE and Spearman's correlation as performance metrics of ordinal classification tasks is that they are influenced by the choice of the numeric label given to the ordinal classes (Cardoso & Sousa, 2011). Other measures have been suggested that are not influenced by the numeric label of ordinal classes (Cardoso & Sousa, 2011), but we have maintained the use of these metrics for this study for ease of computation and readers' familiarity.

4.2 Conclusions

This simulation study highlighted the variability in relative predictive performance of common treatments of ordinal outcome variables in machine learning according to key factors, including the algorithm being used to implement the approaches and the degree of class imbalance in the ordinal outcome. The consideration of these important, practical factors extends the previous literature on ordinal classification and provides further knowledge on the consequences of naïve treatments of ordinal outcomes, which are shown to vary substantially according to these factors. Given the ubiquity of ordinal measures in the behavioral sciences coupled with the growing use of machine learning in the behavioral sciences, these are important considerations for building high-performing models in the field.

References

Baccianella, S., Esuli, A., & Sebastiani, F. (2009). Evaluation measures for ordinal regression. In *2009 ninth international conference on intelligent systems design and applications.* IEEE. doi: https://doi.org/10.1109/isda.2009.230

Ben-David, A., Sterling, L., & Tran, T. (2009, apr). Adding monotonicity to learning algorithms may impair their accuracy. *Expert Systems with Applications*, *36*(3), 6627–6634. doi: https://doi.org/10.1016/j.eswa.2008.08.021

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. doi: https://doi.org/10.1023/a:1010933404324

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (2017). *Classification and regression trees.* Routledge. doi: https://doi.org/10.1201/9781315139470

Bürkner, P.-C., & Vuorre, M. (2019, feb). Ordinal regression models in psychology: A tutorial. *Advances in Methods and Practices in Psychological Science*, *2*(1), 77–101. doi: https://doi.org/10.1177/2515245918823199

Cardoso, J. S., & da Costa, J. F. P. (2007). Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research*, *8*(50), 1393–1429. Retrieved from `http://jmlr.org/papers/v8/cardoso07a.html`

Cardoso, J. S., & Sousa, R. (2011, dec). Measuring the performance of ordinal classification. *International Journal of Pattern Recognition and Artificial Intelligence*, *25*(08), 1173–1195. doi: https://doi.org/10.1142/s0218001411009093

Cheng, J., Wang, Z., & Pollastri, G. (2008, jun). A neural network approach to ordinal regression. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence).* IEEE. doi: https://doi.org/10.1109/ijcnn.2008.4633963

Chu, W., & Keerthi, S. S. (2007, mar). Support vector ordinal regression. *Neural Computation*, *19*(3), 792–815. doi: https://doi.org/10.1162/neco.2007.19.3.792

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Routledge. doi: https://doi.org/10.4324/9780203771587

Crammer, K., & Singer, Y. (2005, jan). Online ranking by projecting. *Neural Computation*, *17*(1), 145–175. doi: https://doi.org/10.1162/0899766052530848

Deng, W.-Y., Zheng, Q.-H., Lian, S., Chen, L., & Wang, X. (2010, dec). Ordinal extreme learning machine. *Neurocomputing*, *74*(1-3), 447–456. doi: https://doi.org/10.1016/j.neucom.2010.08.022

Fernandez-Navarro, F., Riccardi, A., & Carloni, S. (2014, nov). Ordinal neural networks without iterative tuning. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(11), 2075–2085. doi: https://doi.org/10.1109/tnnls.2014.2304976

Frank, E., & Hall, M. (2001). A simple approach to ordinal classification. In *Machine learning: ECML 2001* (pp. 145–156). Springer Berlin Heidelberg. doi: https://doi.org/10.1007/3-540-44795-4_13

Gaudette, L., & Japkowicz, N. (2009). Evaluation methods for ordinal classification. In *Advances in artificial intelligence* (pp. 207–210). Springer Berlin Heidelberg. doi: https://doi.org/10.1007/978-3-642-01818-3_25

Gu, B., Sheng, V. S., Tay, K. Y., Romano, W., & Li, S. (2015, jul). Incremental support vector learning for ordinal regression. *IEEE Transactions on Neural Networks and Learning Systems*, *26*(7), 1403–1416. doi: https://doi.org/10.1109/tnnls.2014.2342533

Gutierrez, P. A., Perez-Ortiz, M., Sanchez-Monedero, J., Fernandez-Navarro, F., & Hervas-Martinez, C. (2016, jan). Ordinal regression methods: Survey and experimental study. *IEEE Transactions on Knowledge and Data Engineering*, *28*(1), 127–146. doi: https://doi.org/10.1109/tkde.2015.2457911

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning.* Springer New York. doi: https://doi.org/10.1007/978-0-387-84858-7

Herbrich, R., Graepel, T., & Obermayer, K. (2000). Large margin rank boundaries for ordinal regression. In P. J. Bartlett, B. Schölkopf, D. Schuurmans, & A. J. Smola (Eds.), *Advances in large margin classifiers* (pp. 115–132). MIT Press.

Hühn, J. C., & Hüllermeier, E. (2008). Is an ordinal class structure useful in classifier learning? *International Journal of Data Mining, Modelling and Management*, *1*(1), 45. doi: https://doi.org/10.1504/ijdmmm.2008.022537

Kramer, S., Widmer, G., Pfahringer, B., & de Groeve, M. (2000). Prediction of ordinal classes using regression trees. In *Lecture notes in computer science* (pp. 426–434). Springer Berlin Heidelberg. doi: https://doi.org/10.1007/3-540-39963-1_45

Kuhn, M. (2022). caret: Classification and regression training [Computer software manual]. Retrieved from `https://github.com/topepo/caret/` (R package version 6.0-93)

Leisch, F., & Dimitriadou, E. (2021). *mlbench: Machine learning benchmark problems.* Retrieved from `https://cran.r-project.org/package=mlbench` (R package version 2.1-3.)

Liaw, A., & Wiener, M. (2002). Classification and regression by randomForest. *R News*, *2*(3), 18–22. Retrieved from `http://CRAN.R-project.org/doc/Rnews/`

Liddell, T. M., & Kruschke, J. K. (2018, nov). Analyzing ordinal data with metric models: What could possibly go wrong? *Journal of Experimental Social Psychology*, *79*, 328–348. doi: https://doi.org/10.1016/j.jesp.2018.08.009

Lin, H.-T., & Li, L. (2012, may). Reduction from cost-sensitive ordinal ranking to weighted binary classification. *Neural Computation*, *24*(5), 1329–1367. doi: https://doi.org/10.1162/neco_a_00265

R Core Team. (2022). *R: a language and environment for statistical computing.* Vienna, Austria: R Foundation for Statistical Computing. Retrieved from `http://www.r-project.org`

Strobl, C., Malley, J., & Tutz, G. (2009, dec). An introduction to recursive partitioning: Rationale, application, and characteristics of classification and regression trees, bagging, and random forests. *Psychological Methods*, *14*(4), 323–348. doi: https://doi.org/10.1037/a0016973

Therneau, T., & Atkinson, B. (2022). rpart: Recursive partitioning and regression trees [Computer software manual]. Retrieved from `https://CRAN.R-project.org/package=rpart` (R package version 4.1.16)

Appendix A Sample R Code

Below, we provide sample R code to demonstrate the implementation of the FH approach using CART and random forests on an applied dataset. The dataset (N = 1,014) contains predictors of maternal health risk among pregnant patients, including age, systolic and diastolic blood pressure, blood glucose levels, body temperature, and heart rate. The task is to predict maternal mortality risk level, an ordinal outcome variable with $k = 3$ classes of low, mid, and high risk. These classes are distributed in the following manner: low risk (40.0%), mid risk (33.1%), and high risk (26.8%). The dataset is publicly available from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Maternal+Health+Risk+Data+Set). Note that the code can easily be adapted to implement the FH approach using other algorithms besides CART and random forests and on datasets with different values of k. The models are trained using the *caret* package, which streamlines the model training and hyperparameter tuning processes and provides a unified syntax for fitting different algorithms.

The dataset is saved in a `data.frame` object called `data`. The outcome variable is a column in `data` called `RiskLevel`.

First, we code the outcome variable as an ordinal variable. We save the class labels and the number of classes into respective objects to be referenced throughout the rest of the program.

```
data$RiskLevel = factor(data$RiskLevel,
                        levels = c("low risk",
                        "mid risk", "high risk"),
                        ordered = TRUE)
classes = levels(data$RiskLevel)
k = length(classes)
```

Next, we split the dataset into a training (50%) and test (50%) set.

```
set.seed(12345)
train = sample(seq_len(nrow(data)),
           size = floor(nrow(data)/2), replace = FALSE)
dtrain = data[train, ]
dtest = data[-train, ]
```

To implement the FH approach, we generate $k - 1$ modified copies of the training set and save each one into a list called `dtrain_modified`. The j^{th} training set has a modified outcome variable that is a binary indicator for whether the original outcome is greater than the j^{th} class. The predictors remain unchanged.

```
dtrain_modified = list()
for (j in 1:(k-1)){
  dt = dtrain
  dt$RiskLevel = as.factor(ifelse(dtrain$RiskLevel >
                                  classes[j], 1, 0))
  dtrain_modified[[j]] = dt
}
```

We initialize two $N_{testset}$ by $(k-1)$ matrices, one for each algorithm, to store the predicted probabilities from the $k-1$ binary classifiers.

```
probsCART = probsRF = matrix(ncol = k-1,
                             nrow = nrow(dtest))
```

We are using the *caret* package to train the CART and random forest models. Below, we set up a control parameter for conducting 5-fold cross validation during training for hyperparameter tuning.

```
library(caret)
trnCntrl = trainControl(method ='cv', number = 5)
```

Below is the for-loop where we train the $k-1$ binary classifiers per algorithm. In the j^{th} iteration (there are $k-1$ iterations) of the loop, we train a CART model and a random forest model on the j^{th} modified training set. After training each model per iteration, we obtain predicted probabilities on the test set and save them into the j^{th} column of the matrix we initialized above. To use a different algorithm, simply change the `method` argument inside of the `train` function.

```
for (j in 1:(k-1)){
  # CART
  modCART_j = train(RiskLevel ~ .,
                    data = dtrain_modified[[j]],
                    method = 'rpart',
                    tuneLength = 10,
                    trControl = trnCntrl)
  pred = predict(modCART_j$finalModel, dtest,
                    type = "prob")[,"1"]
  probsCART[,j] = pred

  # random forest
  modRF_j = train(RiskLevel ~ .,
                  data = dtrain_modified[[j]],
                  method = 'rf',
                  tuneLength = 5,
                  trControl = trnCntrl)
  pred = predict(modRF_j$finalModel, dtest,
                    type = "prob")[,"1"]
  probsRF[,j] = pred

}
```

Next, we combine the $k - 1$ predicted probabilities from the $k - 1$ binary classifiers to obtain predicted probabilities for each of the k classes. We initialize two $N_{testset}$ by k matrices, one for each algorithm, to store these probabilities.

```
probsCART_k = probsRF_k = data.frame(matrix(ncol = k,
                          nrow = nrow(dtest)))
colnames(probsCART_k) = colnames(probsRF_k) = classes
```

The $k - 1$ predicted probabilities are combined according to the rules described in the Methods section of the study to obtain the k predicted probabilities.

```
probsCART_k[,1] = 1 - probsCART[, 1]
probsCART_k[,k] = probsCART[, (k-1)]

probsRF_k[,1] = 1 - probsRF[, 1]
probsRF_k[,k] = probsRF[, (k-1)]

for (i in 2:(k-1)){
  probsCART_k[,i] = probsCART[, (i-1)] - probsCART[, i]
  probsRF_k[,i] = probsRF[, (i-1)] - probsRF[, i]
}
```

Finally, each test set observation is assigned to the ordinal class with the largest predicted probability. We store the predicted class labels into a vector for each algorithm, which can be used to compute overall performance metrics, such as the mean absolute error and Spearman's correlation.

```
predclassCART = colnames(probsCART_k)[max.col(probsCART_k,
                        ties.method = "random")]
predclassCART = factor(predclassCART, levels = classes)

predclassRF = colnames(probsRF_k)[max.col(probsRF_k,
                        ties.method = "random")]
predclassRF = factor(predclassRF, levels = classes)

# Mean absolute error
mean(abs(as.integer(predclassCART)
         - as.integer(dtest$RiskLevel)))
mean(abs(as.integer(predclassRF)
         - as.integer(dtest$RiskLevel)))

# Spearman
cor(as.integer(predclassCART), as.integer(dtest$RiskLevel),
    method = "spearman")
cor(as.integer(predclassRF), as.integer(dtest$RiskLevel),
    method = "spearman")
```

Class-level performance can conveniently be obtained using the `confusionMatrix` function of the *caret* package.

```
# class-level performance
cmCART = confusionMatrix(predclassCART, dtest$RiskLevel)
cmRF = confusionMatrix(predclassRF, dtest$RiskLevel)

cmCART$byClass
cmRF$byClass
```

In the table below, we summarize the empirical performance results of the FH approach using CART and random forests on this applied dataset, along with the performance of the naïve classification and naïve regression approaches. Naïve classification and naïve regression can be implemented by simply converting `data$RiskLevel` into an unordered factor and into an integer, respectively. These results show that the FH approach generally outperformed both naïve approaches, although the improvement is minimal, especially for the random forest implementation.

	CART		
Performance metric	FH	Naïve classification	Naïve regression
Overall performance			
Spearman's correlation	.73	.67	.69
Mean absolute error	.30	.35	.33
Misclassification rate	.29	.32	.33
Class-level performance (F1)			
Low risk class	.73	.70	.68
Mid risk class	.57	.53	.58
High risk class	.85	.83	.80
	Random forest		
	FH	Naïve classification	Naïve regression
Overall performance			
Spearman's correlation	.80	.80	.78
Mean absolute error	.20	.22	.24
Misclassification rate	.19	.20	.23
Class-level performance (F1)			
Low risk class	.82	.81	.76
Mid risk class	.75	.71	.70
High risk class	.90	.89	.88

Appendix B Additional Results

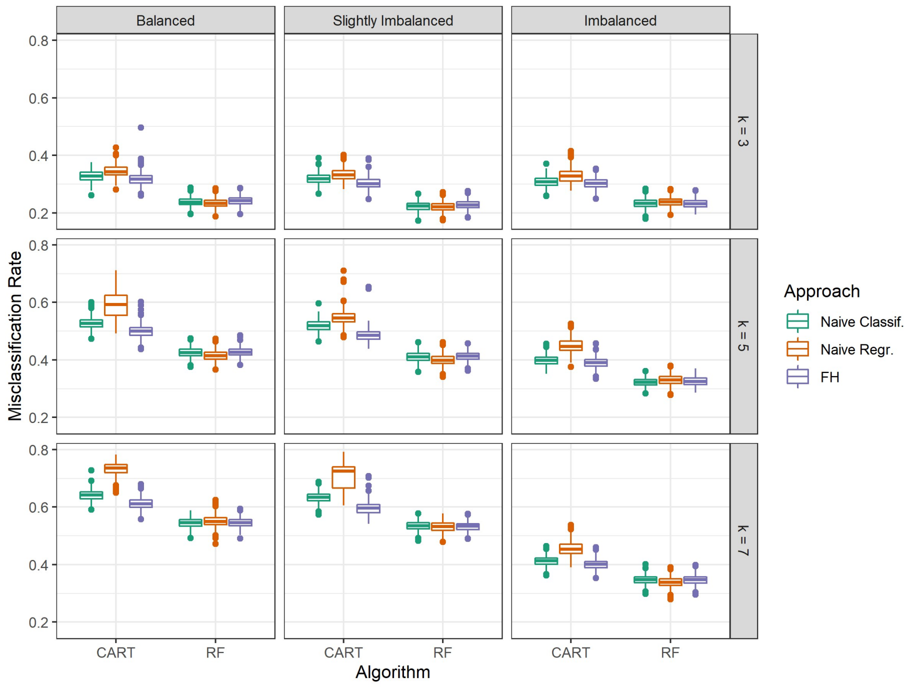


Figure B.1: Distribution of misclassification rate of the six models across replications in each simulation condition

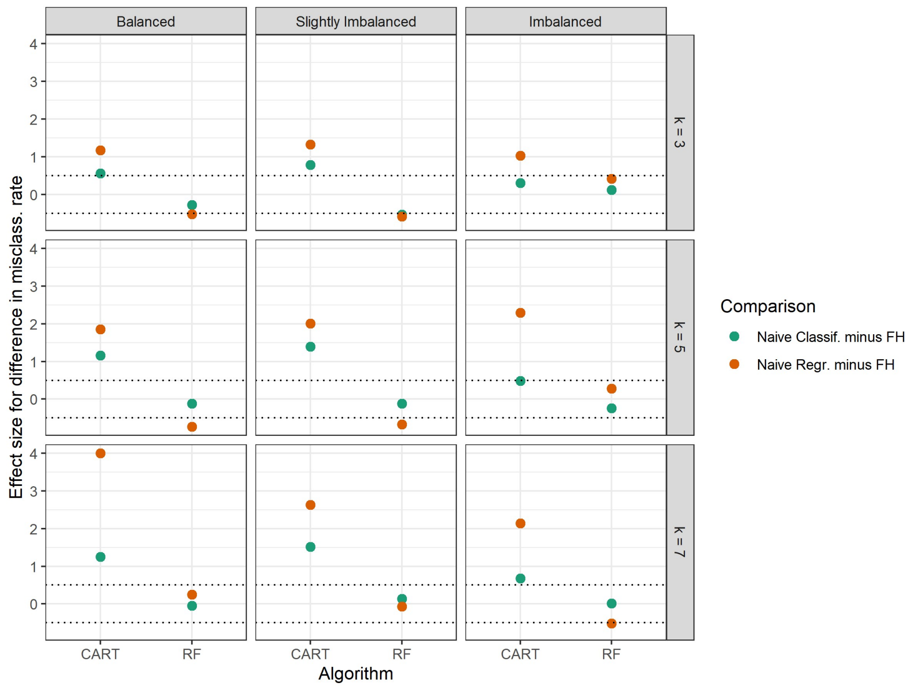


Figure B.2: Effect sizes for the paired difference in misclassification rate between the FH approach and naïve approaches for each algorithm for each simulation condition

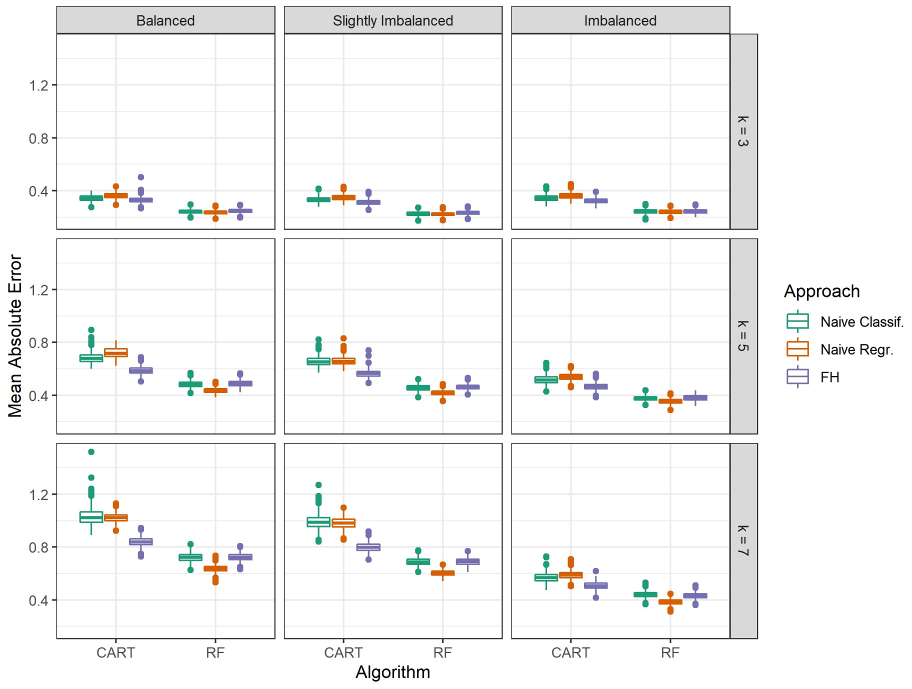


Figure B.3: Distribution of mean absolute error of the six models across replications in each simulation condition

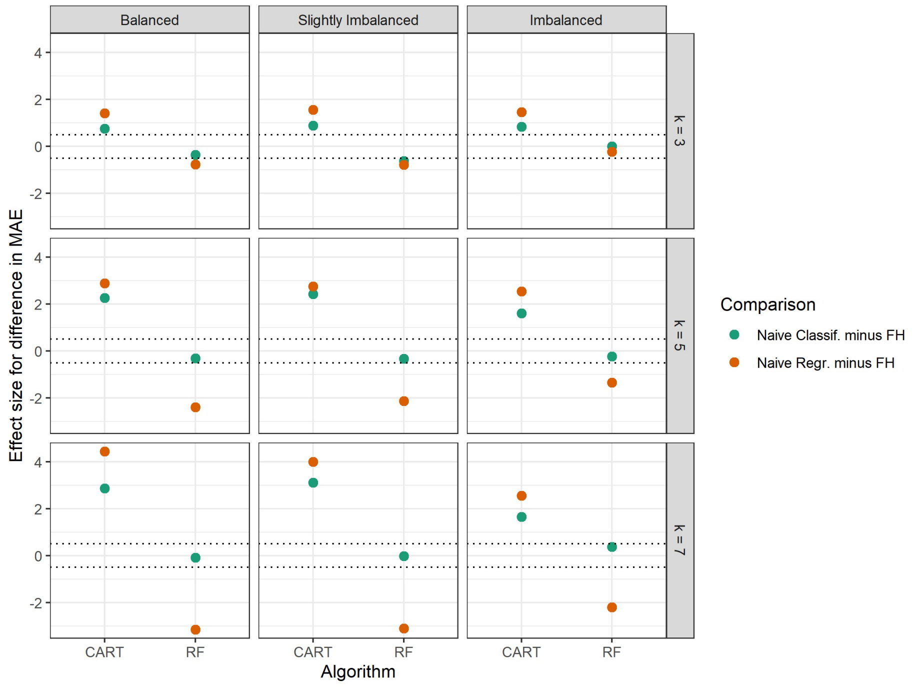


Figure B.4: Effect sizes for the paired difference in mean absolute error between the FH approach and naïve approaches for each algorithm for each simulation condition

Journal of Behavioral Data Science, 2022, *2* (2), 99–126.
DOI: `https://doi.org/10.35566/jbds/v2n2/xu`

Handling Ignorable and Non-ignorable Missing Data through Bayesian Methods in JAGS

Ziqian Xu[1]

University of Notre Dame, Notre Dame, IN 46530, USA
`zxu9@nd.edu`

Abstract. With the prevalence of missing data in social science research, it is necessary to use methods for handling missing data. One framework in which data with missing value can still be used for parameter estimation is the Bayesian framework. In this tutorial, different missing data mechanisms including Missing Completely at Random, Missing at Random, and Missing Not at Random are introduced. Methods for estimating models with missing values under the Bayesian framework for both ignorable and non-ignorable missingness are also discussed. A structural equation model on data from the Advanced Cognitive Training for Independent and Vital Elderly study is used as an illustration on how to fit missing data models in JAGS.

Keywords: Missing data · Bayesian analysis· Structural equation model

1 Introduction

The problem of missing data is prevalent in research, and the social sciences are particularly influenced by this problem because surveys are commonly used to collect information. As pointed out by Berchtold (2019), item-level missing data were found in 69.5% of papers published in selected social science journals, suggesting that the issue of missing data is quite omnipresent. Missing data may lead to various problems including biases in estimations and lowered identifiability of models (e.g., Zhang & Wang, 2012). To address missing data issues, procedures including listwise deletion, full-information maximum likelihood estimation, and multiple imputation have been proposed (e.g., Zhang & Wang, 2013). One emerging context where models can be fitted with the presence of missing data is the Bayesian framework (Ma & Chen, 2018). With Bayesian inference, missing data can be handled quite naturally. The purpose of this paper is to demonstrate the procedure of fitting a structural equation model with missing data under the Bayesian framework using the software JAGS (Plummer et al., 2003).

1.1 Missing Data Mechanisms

Three missing data mechanisms, Missing Completely at Random (MCAR), Missing at Random (MAR), and Missing Not at Random (MNAR), have been discussed in the literature (Rubin, 1976). To distinguish these mechanisms, we can use a vector Y to denote the outcome variable, a matrix X to denote the covariates, and a vector R to denote the missing data indicator for the outcome Y, with $R = 1$ if a Y is missing, otherwise 0. We can also assume a model with parameter γ that governs the generation process of R. Further, we can assume that the purpose of data analysis is to obtain the parameter θ that explains Y using X. To illustrate the different missing data mechanisms, we use missingness in the outcome Y as an example.

- If the missing data in the outcome Y are MCAR, the missing data do not depend on any data collected. Thus, $P(R|\gamma, Y, X) = P(R|\gamma)$.
- If the missing data in the outcome Y are MAR, the missingness depends on the data collected, thus, $P(R|\gamma, Y, X) = P(R|\gamma, Y_{obs}, X_{obs})$ where X_{obs} denotes covariates that are observed and used in the modeling processes to obtain θ, and Y_{obs} denotes the observed part of the outcome.
- If the missing data in the outcome Y are MNAR, the missingness depends on the unobserved data, thus, $P(R|\gamma, Y, X) = P(R|\gamma, Y_{mis}, X_{mis}, Y_{obs}, X_{obs})$ where X_{mis} denotes predictors or covariates that are not observed and Y_{mis} denotes the missing outcome data.

1.2 Bayesian Methods for Handling Missing Outcome Data

Ignorable Missing Data Under the Bayesian framework, when the missing outcome data mechanism is ignorable (MCAR or MAR), meaning that the missingness can be viewed as random after accounting for observed data, the parameter θ of interest in the statistical analysis model can be estimated based on the observed part of data Y_{obs}. For simplicity, we assume covariates X are fully observed for now. Thus, for ignorable missing outcome data, we can derive $P(\theta|Y_{obs}, X) \propto P(Y_{obs}|\theta, X)\pi(\theta)$ as the posterior of θ based on the observed part of data $P(Y_{obs}|\theta, X)$ and the prior $\pi(\theta)$. Any suitable model for the outcome variable can be used under the above scheme to obtain posterior samples of the parameter when missing data are present (Ma & Chen, 2018).

Non-ignorable Missing Data When the missing outcome data are non-ignorable (MNAR), meaning that the missingness mechanism cannot be fully explained by observed data, models such as the selection model and the pattern-mixture model can be used. Again, we assume that the covariates X are fully observed for now. When missing data are non-ignorable, the target parameter of estimation is not only θ, but both θ and γ.

Selection Model. The selection model partitions the joint conditional probability of the outcome variable Y and missing data indicator R into two parts: $P(Y, R|\theta, \gamma, X) = P(Y|\theta, X) \cdot P(R|\gamma, Y, X)$ (Heckman, 1979). The part $P(Y|\theta, X)$ denotes the probability of the outcome Y based on covariates X and parameters θ not accounting for missingness. The part $P(R|\gamma, Y, X)$ denotes the missingness mechanism based on both the covariates and the outcome data. While the selection model assumes the same analysis model for observed and missing data, it also assumes that the missingness indicator can be viewed as a function of the data. To incorporate the selection model into the Bayesian estimation process, we can write the posterior as $P(\theta, \gamma|Y, X, R) \propto P(Y|\theta, X)P(R|\gamma, X, Y)\pi(\theta, \gamma)$ so that in addition to the the model parameter θ, the missing data parameter γ is also estimated.

Pattern-Mixture Model. The pattern-mixture model factors the joint conditional probability of the outcome Y and the missingness indicator R in a different way: $P(Y, R|\theta, \gamma, X) = P(Y|\theta, X, R) \cdot P(R|\gamma, X)$ (Little, 1994). In this model , the part $P(Y|\theta, X, R)$ indicates that the outcome Y depends on missingness (i.e., the outcome model could be different for observed and missing data resembling a mixture model), and the part $P(R|\gamma, X)$ denotes the missingness mechanisms that only depend on the covariates and not on the outcome. To incorporate the pattern-mixture model into the Bayesian estimation process, we can express the posterior as $P(\theta, \gamma|Y, X, R) \propto P(Y|\theta, X, R)P(R|\gamma, X)\pi(\theta, \gamma)$.

1.3 Bayesian Methods for Handling of Missing Covariates Data

In Section 1.2, Bayesian schemes for handing missing data in the outcome variable are discussed assuming that the covariates are complete. However, covariates often contain missing data as well in real life. Thus, the estimation procedures need further adaptations. When the missing data mechanism for the covariates are ignorable (MCAR or MAR), distributions of the covariates can be specified in addition to the models in Section 1.2 such that each covariate can have its own conditional distribution (Ibrahim, Chen, & Lipsitz, 2002). When the missing covariates are non-ignorable (MNAR), models similar to the non-ignorable missing outcome model can also be applied to the covariates such as the selection model for both outcome and covariates implemented by Lee and Tang (2006).

2 Data and Models

2.1 Data

A subset of data from the The Advanced Cognitive Training for Independent and Vital Elderly (ACTIVE) study will be used to illustrate how to handle missing data under the Bayesian framework. $n = 500$ records for 5 variables, including 3 items measuring reasoning ability (i.e., *WS, LS*, and *LT*), *AGE*, and *EPT* will be used. Here, *WS* is the word series test result, *LS* is the letter series test result,

LT is the letter sets test result, AGE is the age of participants, and EPT is the Everyday Problems Test result. The respective scores are the integer number of correct answers in each test.

The subset of data selected contains no missing values. For the purpose of this tutorial, missing values are generated according to the MAR and MNAR mechanisms. For the ignorable missingness model, 14.8% of missing data in EPT are simulated such that $logit(P(R_i = 1)) = -0.6 + 0.01 * Age_i$ where the R_i is the missingness indicator for the i-th record on EPT. For the selection model on handling non-ignorable missingness, 16.2% of missing data in EPT are simulated such that $logit(P(R_i = 1)) = 0.01 * EPT_i$. Lastly, for the pattern mixture model on handing non-ignorable missingness, 17.6% of missing data in EPT are simulated by viewing EPT values between 24 and 26 as missing. Because EPT consists of integer values only, this means that only the EPT values of 24, 25 and 26 are missing. Thus, when viewing EPT as a linear function of other variables, which is what we will use in the analysis, the slopes for predictors will be small for missing EPT since there is very low variance in missing EPT compared to other variables; whereas for observed EPT, we can expect larger slopes. Additionally, because EPT values are mostly below 24, we can also expect the intercept for missing EPT to be larger than that of observed EPT.

2.2 Analysis Model

The analysis model will be a structural equation model using the three reasoning ability items (i.e., *WS, LS*, and LT) and AGE to predict EPT as shown in Figure 1. The measurement model can be written as in Equation 1:

$$\mathbf{X_m} = \boldsymbol{\mu} + \boldsymbol{\eta}\boldsymbol{\Lambda} + \boldsymbol{\epsilon}. \tag{1}$$

We denote the three manifest variables that are the reasoning ability items involved in the measurement model using the matrix $\mathbf{X_m}$ with dimension $n \times 3$ as formatted in Equation 2. In addition, the latent variable $REASON$ is represented by the second column in $\boldsymbol{\eta}_{n\times1}$, the factor loadings are represented by $\boldsymbol{\Lambda}_{1\times3}$, the intercepts are represented by $\boldsymbol{\mu}_{1\times3}$, and the error terms are represented by $\boldsymbol{\epsilon}_{n\times3}$. In the measurement model, it is assumed that $\epsilon_{i1} \sim \mathcal{N}(0, \sigma^2_{WS})$, $\epsilon_{i2} \sim \mathcal{N}(0, \sigma^2_{LT})$, and $\epsilon_{i3} \sim \mathcal{N}(0, \sigma^2_{LS})$ for $i \in (1, ..., n)$. Further, we assume the latent variable $REASON_i \sim \mathcal{N}(0, \sigma^2_{REASON})$.

$$
\begin{aligned}
\mathbf{X_m} &= \begin{bmatrix} WS_1 & LT_1 & LS_1 \\ ... & ... & ... \\ WS_n & LT_n & LS_n \end{bmatrix} \\
\boldsymbol{\mu} &= \begin{bmatrix} k_1 & k_2 & k_3 \end{bmatrix} \\
\boldsymbol{\eta} &= \begin{bmatrix} REASON_1 \\ ... \\ REASON_n \end{bmatrix} \\
\boldsymbol{\Lambda} &= \begin{bmatrix} 1 & \lambda_{LT} & \lambda_{LS} \end{bmatrix} \\
\boldsymbol{\epsilon} &= \begin{bmatrix} \epsilon_{11} & \epsilon_{12} & \epsilon_{13} \\ ... & ... & ... \\ \epsilon_{n1} & \epsilon_{n2} & \epsilon_{n3} \end{bmatrix}
\end{aligned}
\quad . \tag{2}
$$

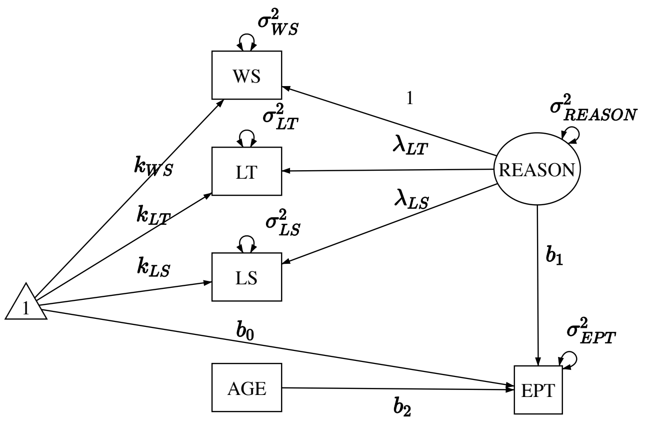


Figure 1. Analysis model for the ACTIVE data.

The structural model follows Equation 3. Since the outcome is one-dimensional, we write the structural model with respect to the outcome EPT directly. In the structural model, the latent variable $REASON$ and the observed variable AGE are used to predict EPT with an intercept. Similar to the assumptions of the measurement model, it is assumed that $\epsilon_i^{EPT} \sim \mathcal{N}(0, \sigma^2_{EPT})$.

$$EPT_i = b_0 + b_1 REASON_i + b_2 AGE_i + \epsilon_i^{EPT}. \tag{3}$$

Together, the measurement and structural models can also be written with respect to each item as in Equation 4:

$$\begin{aligned} WS_i &= k_{WS} + REASON_i + \epsilon_{i1} \\ LT_i &= k_{LT} + \lambda_{LT} REASON_i + \epsilon_{i2} \\ LS_i &= k_{LS} + \lambda_{LS} REASON_i + \epsilon_{i3} \\ EPT_i &= b_0 + b_1 REASON_i + b_2 AGE_i + \epsilon_i^{EPT} \end{aligned} . \qquad (4)$$

Combining the measurement model and the structural model, the probability distribution function form of the analysis model used can be written as in Equation 5:

$$\begin{aligned} REASON_i &\sim \mathcal{N}(0, \sigma^2_{REASON}) \\ WS_i &\sim \mathcal{N}(k_{WS} + REASON_i, \sigma^2_{WS}) \\ LT_i &\sim \mathcal{N}(k_{LT} + \lambda_{LT} REASON_i, \sigma^2_{LT}) \\ LS_i &\sim \mathcal{N}(k_{LS} + \lambda_{LS} REASON_i, \sigma^2_{LS}) \\ EPT_i &\sim \mathcal{N}(b_0 + b_1 REASON_i + b_2 AGE_i, \sigma^2_{EPT}) \end{aligned} . \qquad (5)$$

2.3 Software

JAGS will be used to conduct Bayesian inference in this paper. JAGS can be installed via `https://mcmc-jags.sourceforge.io/` (Plummer et al., 2003). In this paper, JAGS will be used in the R environment (R Core Team, 2021) through the software RStudio (RStudio Team, 2022) and the package *runjags* (Denwood, 2016). Other packages such as *rjags* (Plummer, Stukalov, & Denwood, 2022) that provide similar functionalities to *runjags* are available as well. JAGS can also be executed from the command line without using another interface.

In R, the package *runjags* can be installed using `install.packages("runjags")` and loaded using `library(runjags)`.

3 Ignorable Missingness Model

3.1 Model Specification

In Bayesian inference with JAGS, the two crucial components to a model are the likelihood and the priors.

Likelihood If the missingness in the outcome of EPT is ignorable in the data, then missing data can be addressed by sampling from the analysis model using observed data. Thus, the model in Equation 4 can be used as the ignorable missingness model. Further, the likelihood can be specified using the probability densities in Equation 5. In this example, we do not have missing data in the covariates. However, when missing data are present in the covariates, additional distributions will need to be specified for them.

Priors We also need to specify the priors for the ignorable missingness model. Since we are using the assumption that the observed and latent variables all follow normal distributions, which is common in the practice of structural equation modeling, we can choose the priors in Equation 6 for the model parameters. In particular, the regression coefficients, factor loadings, and intercepts will have normal priors, and the variance components will have inverse gamma priors. Because there is only one latent factor in the model used, the inverse gamma distribution could satisfy as the prior for the factor variance. However, when more than one factors are involved in a structural equation model, and when the factors are allowed to correlate, the inverse wishart distribution can be used as the prior for the factor covariance matrix.

$$\begin{aligned} b_0, b_1, b_2 &\sim \mathcal{N}(\mu_b, \sigma_b^2) \\ \sigma^2_{EPT} &\sim \mathcal{IG}(h_{EPT}, \theta_{EPT}) \\ \lambda_{LT}, \lambda_{LS} &\sim \mathcal{N}(\mu_\lambda, \sigma_\lambda^2) \\ \sigma^2_{WS}, \sigma^2_{LT}, \sigma^2_{LS} &\sim \mathcal{IG}(h_m, \theta_m) \\ k_{WS}, k_{LT}, k_{LS} &\sim \mathcal{N}(\mu_k, \sigma_k^2) \\ \sigma^2_{REASON} &\sim \mathcal{IG}(h_{REASON}, \theta_{REASON}) \end{aligned} \quad . \tag{6}$$

In this tutorial, we will choose rather uninformative priors. But more informative priors can also be adopted. For example, the power priors can be used which constructs priors based on likelihood in historical data (Ibrahim & Chen, 2000). Hierarchical priors is another option which can be used when a range of values for the priors are available, so different levels of priors can be specified (Berger & Strawderman, 1996). These methods can be utilized for more theory-informed specification of priors.

3.2 Implementation in JAGS

Implementing a model in JAGS involves the following steps.

Model Specification in JAGS Using *runjags*, the model specification can be stored as a string. For example, the code below specifies a model named `ignorable`.

```
ignorable <- "
#likelihood
...
#prior
...
"
```

The likelihood as specified in Section 3.1 can be implemented inside the model string as the following. In JAGS, distributions can be specified using the symbol ~ . As an illustration, in the code `EPT[i] ~ dnorm(mu.EPT[i], pre.EPT)`, `dnorm` indicates that *EPT* is expected to follow a normal distribution with the mean of `mu.EPT` and the precision (i.e., inverse of variance) of `pre.EPT`. Further, `mu.EPT[i] <- b0 + b1*REASON[i] + b2*AGE[i]` shows that *EPT* is predicted by the latent variable *REASON* and the observed variable *AGE*. The two lines of code together correspond to Equation 3. Note that while `~` is used to specify a distribution, `<-` is used to assign values. Similar to how the distribution of *EPT* is specified, the measurement model for the 3 reasoning ability items can be specified.

```
# likelihood
for (i in 1:N){
  EPT[i] ~ dnorm(mu.EPT[i], pre.EPT)
  mu.EPT[i] <- b0 + b1*REASON[i] + b2*AGE[i]
  WS[i] ~ dnorm(mu1[i], pre.WS)
  LT[i] ~ dnorm(mu2[i], pre.LT)
  LS[i] ~ dnorm(mu3[i], pre.LS)
  mu1[i] <- REASON[i] + k.WS
  mu2[i] <- l.LT*REASON[i] + k.LT
  mu3[i] <- l.LS*REASON[i] + k.LS
  REASON[i] ~ dnorm(0, pre.REASON)
}
```

As mentioned in Section 3.1, historical data and previous research conclusions are not used to construct the priors here; instead, the priors used are rather uninformative. For example, the prior for means the error terms in the factor model of *REASON* has precision of 0.001 or variance of 1000.

```
# priors
# regression model
b0 ~ dnorm(0, pre.b)
b1 ~ dnorm(0, pre.b)
b2 ~ dnorm(0, pre.b)
pre.b ~ dgamma(.001,.001)
pre.EPT ~ dgamma(0.001, 0.001)

# factor model
l.LT ~ dnorm(0, 0.001)
l.LS ~ dnorm(0, 0.001)
pre.WS ~ dgamma(0.001, 0.001)
pre.LT ~ dgamma(0.001, 0.001)
pre.LS ~ dgamma(0.001, 0.001)
k.WS ~ dnorm(0, 0.001)
k.LT ~ dnorm(0, 0.001)
k.LS ~ dnorm(0, 0.001)
pre.REASON ~ dgamma(0.001, 0.001)
```

Additionally, to monitor the variances of various variables instead of the precision, we can use the following to convert precisions to variances.

```
# variances
var.EPT <- 1/pre.EPT
var.WS <- 1/pre.WS
var.LT <- 1/pre.LT
var.LS <- 1/pre.LS
var.REASON <- 1/pre.REASON
```

Data Statement and Initial Values To estimate the ignorable missingness model, we define the data to be used as follows.

```
data <- list(N = 500,
             EPT = data_500_mar$EPT,
             AGE = data_500_mar$AGE,
             WS = data_500_mar$WS,
             LT = data_500_mar$LT,
             LS = data_500_mar$LS)
```

We also define initial values for the two MCMC chains that will be used. Slightly different starting values for the two chains are specified to ensure that the two chains will not converge to the same local optimum, if happening, for better convergence diagnostic.

```
inits <- list(list(b0=0, b1=0, b2=0, l.LT=0, l.LS=0,
                   k.WS=0, k.LS=0, k.LT=0,
                   pre.WS=0.1, pre.LS=0.1, pre.LT=0.1,
                   pre.EPT=0.1, pre.REASON=0.1),
              list(b0=1, b1=1, b2=1, l.LT=1, l.LS=1,
                   k.WS=1, k.LS=1, k.LT=1,
                   pre.WS=0.2, pre.LS=0.2, pre.LT=0.2,
                   pre.EPT=0.2, pre.REASON=0.2))
```

Running JAGS for Analysis Finally, the posteriors can be sampled using the `run.jags` command. The `model=ignorable` argument specifies that the model named m1 will be used. The `monitor` argument specifies which parameters or variables we want to sample. The `data=data, n.chains=2, inits=inits` arguments specify the data, number of chains, and initial values to be used, respectively. The arguments `adapt=1000`, `burnin = 3000`, and `sample=100000` specify the number of adaptation, burn-in, and sampling iterations, respectively. Only the sampling iterations will be recorded. The argument `keep.jags.files=FALSE` instructs JAGS to not save any files produced in the sampling process. If the argument is set to `keep.jags.files=FALSE`, then additional files will be saved in a folder named *runjagsfiles*. The argument `thin=1` sets the intervals of recorded samples to be 1, meaning that every iteration will be saved. The argument `method="simple"` indicates that the simple method would be used to compile the model. The argument `tempdir=TRUE` asks JAGS to create a temporary directory to store files instead of saving files in the working directory.

```
out1 <- run.jags(model=ignorable,
                 monitor=c("b0", "b1", "b2",
                "l.LT", "l.LS",
                "k.WS", "k.LT", "k.LS",
                "var.WS", "var.LT",
                "var.LS", "var.EPT", "var.REASON"),
                 data=data, n.chains=2,
                 inits=inits, method="simple",
                 adapt=1000, burnin = 3000,
                 sample=1000000,
                 thin=1,
                 keep.jags.files=FALSE,
                 tempdir=TRUE)
```

3.3 Convergence Diagnostics

The `summary(out1)` command in *runjags* gives the summary statistics of the JAGS output which can be used for convergence diagnostics. We will mainly rely on the Gelman-Rubin test for diagnosing convergence (Gelman & Rubin, 1992). The Gelman-Rubin test statistics, or the potential scale reduction factor, is readily provided by the `summary` function and is suitable when multiple chains are used. As shown in Table 1, the potential scale reduction factors (psrf) for all parameters are below 1.1 and very close to 1, suggesting that the chains have converged. Monte Carlo SEs of most parameters (MCerr) are quite low as well (< 0.05). The effective sample sizes (SSeff) are also acceptable (> 400).

Table 1. Output from the ignorable data model.

	Lower95	Median	Upper95	Mean	MCerr	SSeff	psrf
b0	10.546	16.071	21.450	16.045	0.041	4515	1
b1	0.855	0.960	1.062	0.960	0.000	16217	1
b2	-0.038	0.034	0.110	0.034	0.001	4485	1
l.LT	0.402	0.445	0.489	0.445	0.000	20000	1
l.LS	1.066	1.142	1.222	1.142	0.000	20000	1
k.WS	9.266	9.700	10.129	9.701	0.002	20000	1
k.LT	5.439	5.684	5.926	5.683	0.001	20000	1
k.LS	9.744	10.225	10.726	10.226	0.002	20483	1
var.WS	3.294	4.247	5.265	4.260	0.004	20000	1
var.LT	3.025	3.487	3.995	3.497	0.002	20000	1
var.LS	3.717	4.981	6.230	4.994	0.005	19548	1
var.EPT	12.649	14.831	17.112	14.884	0.008	20000	1
var.REASON	17.037	19.849	23.122	19.903	0.011	20000	1

The `plot(out1)` command can be used to generate trace plots and histograms of the MCMC chains which can further help diagnose the convergence of the target parameters. For example, Figure 2 shows the MCMC plots of the parameter b_1, and the trace plot and histogram both indicate that the parameter converged. The histogram is centered at one mode, and the trace plot looks stable.

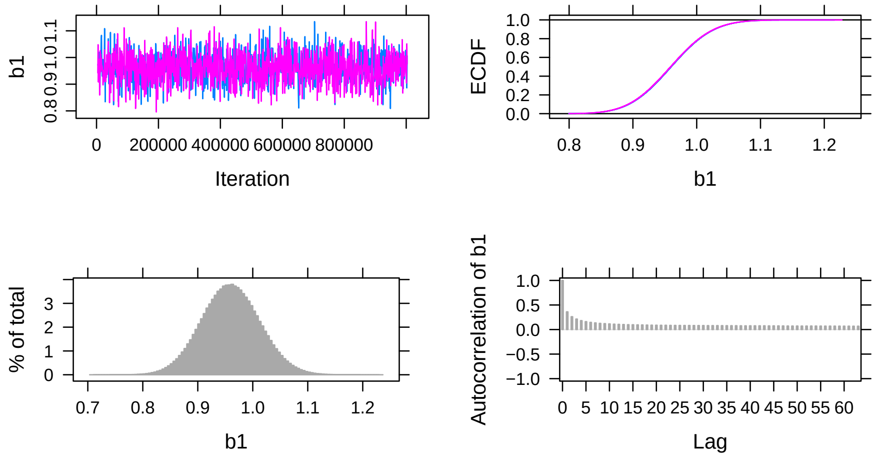


Figure 2. MCMC plots for the parameter b1 in the ignorable missingness model.

There are other approaches for diagnosing convergence which will not be discussed in depth here. For example, the Geweke test can be conducted using the *coda* package (Plummer, Best, Cowles, & Vines, 2006). This test takes two proportions of a Markov chain and applies a z-test to see if the two proportions are different. The Herdelberger and Welch test can also be used with *coda* which tests if the samples come from a stable distribution.

3.4 Interpretation

Using the Highest Posterior Density (HPD) intervals which are indicated by the Lower95 and Upper95 values in Table 1, we can see that the factor loadings and intercepts in the measurement model are likely non-zero as the HPD intervals for them excludes 0. Thus, the items *WS, LT* and *LS* do have commonalities explained by the construct of reasoning ability. In the structural model, the intercept term and the slope for *REASON* have their HPD intervals excluding 0, suggesting that we can conclude that the latent variable of *REASON* explains the variance in *EPT*, whereas *AGE* does not.

4 Selection Model

4.1 Model Specification

Likelihood When the missingness in the outcome is MNAR or non-ignorable, the selection model can be applied. For non-ignorable missing outcome, we also make the same assumption that the missing covariates are ignorable so that we can specify conditional distributions for them. We also assume that the missingness is dependent on the outcome of *EPT* scores themselves in the selection model. Using a missing data indicator R_i where $R_i = 1$ denotes a missing record of *EPT* and $R_i = 0$ denotes an observed record of *EPT*, the selection model translates to Equation 7, which can be used in addition to the model specification in Equation 4:

$$logit(P(R_i = 1)) = a_0 + a_1 EPT_i. \tag{7}$$

The corresponding probability density distribution form of the selection model incorporates Equation 8 in addition to Equation 5:

$$R_i \sim \mathcal{B}(sigmoid(a_0 + a_1 EPT_i)). \tag{8}$$

The missing data indicator R_i is deemed as a function of the outcome value EPT_i and an intercept. The intercept a_0 here is used to adjust the threshold for 1 and 0 in the missing data indicator of R_i.

Priors The priors can be constructed similarly to the ignorable missingness model. Priors for the parameters a_0 and a_1 are needed for the selection model as indicated by Equation 9, which can be combined with the priors in Equation 6 to be used in JAGS.

$$a_0, a_1 \sim \mathcal{N}(\mu_a, \sigma_a^2) \tag{9}$$

4.2 Implementation in JAGS

Model Specification in JAGS In JAGS, the likelihood for the ignorable data model can be modified to incorporate the selection model by including the following.

```
R[i] ~ dbern(p[i])
logit(p[i]) = a0 + a1*EPT[i]
```

The priors also need to be modified based on the the function of missingness on *EPT*.

```
a0 ~ dnorm(0, pre.a)
a1 ~ dnorm(0, pre.a)
pre.a ~ dgamma(.001,.001)
```

Data Statement and Initial Values Since the model has now changed, the initial values and data supplied should be modified accordingly. A missing data indicator for the outcome is also needed as shown below in the variable `R = is.na(data_500_mnar$EPT)*1`.

```
data <- list(N = 500,
             EPT = data_500_mnar$EPT,
             AGE = data_500_mnar$AGE,
             WS = data_500_mnar$WS,
             LT = data_500_mnar$LT,
             LS = data_500_mnar$LS,
             R = is.na(data_500_mnar$EPT)*1)

inits <- list(list(b0=0, b1=0, b2=0, l.LT=0, l.LS=0,
                   k.WS=0, k.LS=0, k.LT=0,
                   pre.WS=0.1, pre.LS=0.1, pre.LT=0.1,
                   pre.EPT=0.1, pre.REASON=0.1,
                   a0=0.1, a1=0.1),
              list(b0=1, b1=1, b2=1, l.LT=1, l.LS=1,
                   k.WS=1, k.LS=1, k.LT=1,
                   pre.WS=0.2, pre.LS=0.2, pre.LT=0.2,
                   pre.EPT=0.2, pre.REASON=0.2,
                   a0=0.2, a1=0.2))
```

Running JAGS for Analysis The model can then be estimated using the following statement. Again, 1000 adaptation iterations, 3000 burn-in iterations, and 100000 sampling iterations are specified.

```
out1 <- run.jags(model=selection,
                 monitor=c("b0", "b1", "b2",
                 "a0", "a1",
                 "l.LT", "l.LS", "k.WS",
                 "k.LT", "k.LS",
                 "var.WS", "var.LT",
                 "var.LS", "var.EPT", "var.REASON"),
                 data=data, n.chains=2,
                 inits=inits, method="simple",
                 adapt=1000, burnin = 3000,
                 sample=1000000,
                 thin=1,
                 keep.jags.files=FALSE,
                 tempdir=TRUE)
```

4.3 Convergence Diagnostics

As show in Table 2, the potential scale reduction factors (psrf) are all below 1.1, and the Monte Carlo SEs (MCerr) are satisfactory (< 0.05). The effective sample sizes (SSeff) for all parameters are also above 400. The trace plots are satisfactory as well. As an example, MCMC plots for the parameter a_1 is shown in Figure 3.

Table 2. Output from the selection model.

	Lower95	Median	Upper95	Mean	MCerr	SSeff	psrf
b0	13.170	18.649	24.029	18.623	0.043	4048	1
b1	0.865	0.968	1.077	0.968	0.000	15546	1
b2	-0.074	-0.002	0.072	-0.002	0.001	4064	1
a0	-4.276	-2.752	-1.374	-2.788	0.006	13872	1
a1	-0.010	0.059	0.132	0.059	0.000	13990	1
l.LT	0.402	0.446	0.491	0.446	0.000	20000	1
l.LS	1.076	1.153	1.233	1.154	0.000	20551	1
k.WS	9.266	9.702	10.133	9.702	0.002	20000	1
k.LT	5.445	5.685	5.926	5.685	0.001	19607	1
k.LS	9.747	10.230	10.720	10.228	0.002	19719	1
var.WS	3.436	4.448	5.453	4.459	0.004	20000	1
var.LT	3.048	3.502	4.014	3.513	0.002	21125	1
var.LS	3.443	4.699	5.955	4.711	0.005	20000	1
var.EPT	12.945	15.249	17.670	15.308	0.009	19667	1
var.REASON	16.618	19.644	22.728	19.701	0.011	20000	1

4.4 Interpretation

Using the HPD intervals, the factor loadings and factor intercepts in the measurement model are again non-zero, indicating that the three items measuring *REASON* do have overlaps. In the structural part of the selection model, the intercept and the slope for the latent variable *REASON* both have HPD intervals excluding 0, suggesting that *REASON* do explain significant variance in *EPT* after accounting for the missingness mechanism. However, the HPD interval for the slope of *AGE* still contains 0, suggesting that *AGE* may not be a useful predictor of *EPT* here.

The slope for *EPT* in explaining the missingness mechanism has HPD intervals containing 0, suggesting that *EPT*'s effect on explaining missingness is small. Whereas the intercept for predicting missingness has its HPD interval excluding 0. This deviates from the data generation model because the coefficients used in missing data generation are quite small.

Although the selection model is most commonly used when missing data are non-ignorable, it can be applied to MCAR and MAR data as well. In the ACTIVE example, missingness predictors can be changed to, for instance, `logit(p[i]) = a0` for MCAR missingness and `logit(p[i]) = a0 + a1*AGE[i]` for MAR missingness, for example.

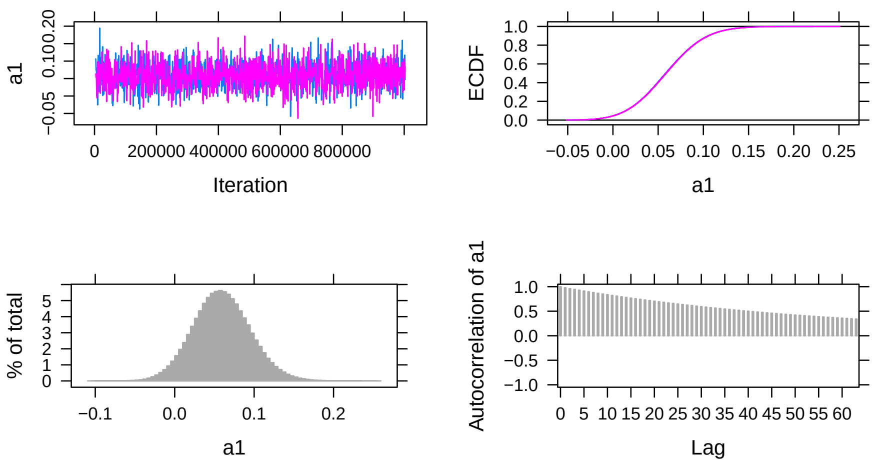


Figure 3. MCMC plots for the parameter a1 in the selection model.

5 Pattern-Mixture Model

5.1 Model Specification

Likelihood A pattern-mixture model can also be used to handle non-ignorable missing data in the outcome of EPT. In this case, the intercept `b0` and the slope `b1` for the outcome regressed on the latent variable of $REASON$ are deemed different for missing and observed data, whereas in practice, theories should be used to guide the decision on the missing data model. The pattern-mixture property is shown as the subscripts R_i on b_0 and b_1 in Equation 10, which can be used in addition to the model specification in Equation 4 as the specification for the pattern-mixture model:

$$EPT_i = b_0^{R_i} + b_1^{R_i} REASON_i + b_2 AGE_i + \epsilon_i^{EPT}. \tag{10}$$

Also, Equation 11 can be used in addition to Equation 5 as the probability density function form of the pattern-mixture model:

$$EPT_i \sim N(b_0^{R_i} + b_1^{R_i} REASON_i + b_2 AGE_i, \sigma_y^2). \tag{11}$$

Assuming $R_i = 1$ for a missing record on EPT and $R_i = 0$ for an observed record on EPT, then there are two sets of distinct b_0 and b_1 values for $R_i = 1$ and

$R_i = 0$, respectively. It should be noted that this is merely one possible pattern-mixture model that can be fitted onto the data, and that other pattern-mixture models can be used based on specific theories.

Priors Priors for the pattern-mixture model can be specified using Equation 12 to replace the corresponding regression coefficients priors in Equation 6. Different from the priors for the ignorable missingness model, here, priors for b_0 and b_1 are differentiated for missing data (b_0^1, b_1^1) and for observed data (b_0^0, b_1^0).

$$b_0^0, b_0^1, b_1^0, b_1^1 \sim \mathcal{N}(\mu_b, \sigma_b^2) \tag{12}$$

5.2 Implementation in JAGS

Model Specification in JAGS Different from the ignorable missingness model, we need to specify the mean of *EPT* to be different for missing and observed data such as `mu.EPT[i] <- b0[R[i]+1] + b1[R[i]+1] *REASON[i] + b2*AGE[i]` in JAGS. Since JAGS uses 1-based indexes instead of 0-based indexes, we need to use `R[i]+1` instead of `R[i]` in the code to transform the 0 and 1 values in the missing value indicator to 1 and 2. We also need to modify the priors of `b0` and `b1` as the following which is based on the assumption that while the intercept is larger for missing *EPT*, the slope is smaller for missing *EPT*. This assumption is based on how missing data is generated.

```
  b0[1] ~ dnorm(0, pre.b) # non-missing
  b0[2] ~ dnorm(b0[1]+05, pre.b) #missing
  b1[1] ~ dnorm(0, pre.b) # non-missing
  b1[2] ~ dnorm(b1[1]-0.5, pre.b) #missing
```

Data Statement and Initial Values Similar to the case in the selection model, an indicator for missing or observed data is required for estimating the pattern-mixture model, and the data statement can take the form of the following:

```
data <- list(N = 500,
             EPT = data_500_mnar_2$EPT,
             AGE = data_500_mnar_2$AGE,
             WS = data_500_mnar_2$WS,
             LT = data_500_mnar_2$LT,
             LS = data_500_mnar_2$LS,
             R = is.na(data_500_mnar_2$EPT)*1)
```

The initial values can be specified based on the pattern-mixture model. Compared to the ignorable missingness model, initial values for `b0` and `b1` are changed to incorporate two cases for missing and observed data. For example, in the first chain, initial values for `b0` and `b1` are `b0=c(0,1), b1=c(1,0)`. This is consistent with our expectation that the intercept and slope for missing data would be greater and lower, respectively, than those of the observed data.

```
inits <- list(list(b0=c(0,1), b1=c(1,0), b2=0,
                   l.LT=0, l.LS=0, k.WS=0,
                   k.LS=0, k.LT=0,
                   pre.WS=0.1, pre.LS=0.1, pre.LT=0.1,
                   pre.EPT=0.1, pre.REASON=0.1),
              list(b0=c(1,2), b1=c(2,1), b2=1,
                   l.LT=1, l.LS=1, k.WS=1,
                   k.LS=1, k.LT=1,
                   pre.WS=0.2, pre.LS=0.2, pre.LT=0.2,
                   pre.EPT=0.2, pre.REASON=0.2))
```

Running JAGS for Analysis Finally, the model can be estimated using the statement below.

```
out1 <- run.jags(model=pmm,
                 monitor=c("b0", "b1", "b2",
                "l.LT", "l.LS",
                "k.WS", "k.LT", "k.LS", "var.WS", "var.LT",
                "var.LS", "var.EPT", "var.REASON"),
                 data=data, n.chains=2,
                 inits=inits, method="simple",
                 adapt=1000, burnin = 3000,
                 sample=1000000,
                 thin=1,
                 keep.jags.files=FALSE,
                 tempdir=TRUE)
```

5.3 Convergence Diagnostics

In Table 3 which contains the output from this model, the potential scale reduction factors (psrf) are lower than 1.1, suggesting that the chains have converged. The effective sample sizes (SSeff) are acceptable (> 400), and the Monte Carlo SEs (MCerr) are mostly below 0.05.

The MCMC plots in Figure 4 can also be used to diagnose convergence. Here, the plot for `b1` when data are observed is used as an example. All parameters, similar to `b1`, present satisfactory trace plots suggesting that convergence is reached and the estimates are stable.

5.4 Interpretation

As shown by the HPD intervals in Table 3, the factor loadings and intercepts in the measurement model again exhibit HPD intervals that exclude 0. In the structural model, *AGE*'s HPD interval contains 0, whereas the intercept and the slope for *REASON* have HPD intervals excluding 0 for observed data and including 0 for missing data. Consequently, for observed data, *REASON* can predict *EPT* whereas for missing data it is not the case.

Table 3. Output from the pattern-mixture model.

	Lower95	Median	Upper95	Mean	MCerr	SSeff	psrf
b0[1]	11.522	16.352	21.157	16.361	0.034	5190	1.00
b0[2]	-7.595	21.045	52.908	21.498	0.113	20000	1.00
b1[1]	0.847	0.950	1.053	0.951	0.000	17588	1.00
b1[2]	-30.761	0.527	29.239	0.586	0.119	20000	1.01
b2	-0.046	0.020	0.084	0.020	0.000	5211	1.00
l.LT	0.399	0.443	0.486	0.443	0.000	20000	1.00
l.LS	1.066	1.141	1.222	1.142	0.000	19925	1.00
k.WS	9.288	9.705	10.149	9.705	0.002	20328	1.00
k.LT	5.450	5.686	5.925	5.686	0.001	19697	1.00
k.LS	9.741	10.232	10.716	10.231	0.002	20291	1.00
var.WS	3.254	4.204	5.240	4.218	0.004	20000	1.00
var.LT	3.070	3.524	4.036	3.533	0.002	20000	1.00
var.LS	3.711	4.955	6.239	4.968	0.005	20000	1.00
var.EPT	11.269	13.222	15.379	13.268	0.007	20000	1.00
var.REASON	16.956	19.852	23.114	19.932	0.011	20000	1.00

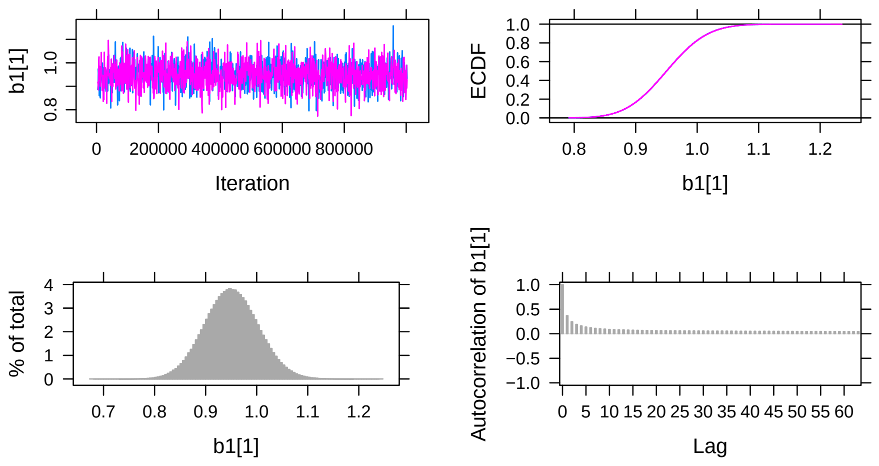


Figure 4. MCMC plots for the parameter b1 for observed *EPT* data in the pattern-mixture model.

6 Discussion

Data with missing values can still be used for parameter estimation in different models under the Bayesian framework. In this paper, a structural equation model is used as an example on the ACTIVE study data to illustrate how models with missing data can be fitted using JAGS in R under different missing data mechanisms including MCAR, MAR, and MNAR. Specifically, two models, the selection model and the pattern-mixture model, are introduced for non-ignorable (i.e., MNAR) missing data.

While this paper mainly focuses on obtaining parameter estimates when missing data are present, other aspects of missing data are not covered here. Although not discussed in this paper, other models for handling non-ignorable missingness exist such as the shared-parameter model (Albert & Follmann, 2008). In addition, there are model selection methods for deciding which of many potential missing data models would suit the data best. For example, in sensitivity analysis which is often used to validate missing data handling processes by assessing robustness of imputations under different conditions, Bayesian model comparison criteria such as the deviation information criterion (DIC) can be used to choose the best model (Ma & Chen, 2018; Van Buuren, 2018). Further, there are ways to diagnose missing data mechanisms in a dataset such as discussed in Little (1988). These topics can be further explored.

There are also different analysis models tailored toward specific types of data that can be incorporated in Bayesian missing data handling processes, which we did not discuss in this paper. While this paper uses a structural equation model as an example, other models can be similarly constructed in JAGS to accommodate missing data. For example, longitudinal data can be analyzed using a multilevel model (Gelman & Hill, 2006). If the response data are of mixed types from mixed populations, then mixture models may be appropriate (Rasmussen, 1999). Social network data can also be used with methods such as the exponential random graph model and the latent space model (Hoff, Raftery, & Handcock, 2002; Robins, Pattison, Kalish, & Lusher, 2007). These models have Bayesian variants that can be applied to estimate parameters when missing data are present (Bürkner, 2017; Koskinen, Robins, Wang, & Pattison, 2013; Neal, 1992).

Acknowledgment

The research reported here was supported, in whole or in part, by the Institute of Education Sciences, U.S. Department of Education, through grant R305D210023 to University of Notre Dame. The opinions expressed are those of the authors and do not represent the views of the Institute or the U.S. Department of Education.

References

Albert, P. S., & Follmann, D. A. (2008). Shared-parameter models. In *Longitudinal data analysis* (pp. 447–466). Chapman and Hall/CRC. doi: https://doi.org/10.1201/9781420011579.ch19

Berchtold, A. (2019). Treatment and reporting of item-level missing data in social science research. *International Journal of Social Research Methodology*, *22*(5), 431–439. doi: https://doi.org/10.1080/13645579.2018.1563978

Berger, J. O., & Strawderman, W. E. (1996). Choice of hierarchical priors: Admissibility in estimation of normal means. *The Annals of Statistics*, 931–951. doi: https://doi.org/10.1214/aos/1032526950

Bürkner, P.-C. (2017). Advanced bayesian multilevel modeling with the r package brms. *arXiv preprint arXiv:1705.11123*. doi: https://doi.org/10.32614/rj-2018-017

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*, 1–25. doi: https://doi.org/10.18637/jss.v071.i09

Gelman, A., & Hill, J. (2006). *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press. doi: https://doi.org/10.1017/cbo9780511790942

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical science*, 457–472. doi: https://doi.org/10.1214/ss/1177011136

Heckman, J. J. (1979). Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, 153–161. doi: https://doi.org/10.2307/1912352

Hoff, P. D., Raftery, A. E., & Handcock, M. S. (2002). Latent space approaches to social network analysis. *Journal of the american Statistical association*, *97*(460), 1090–1098. doi: https://doi.org/10.1198/016214502388618906

Ibrahim, J. G., & Chen, M.-H. (2000). Power prior distributions for regression models. *Statistical Science*, 46–60. doi: https://doi.org/10.1214/ss/1009212673

Ibrahim, J. G., Chen, M.-H., & Lipsitz, S. R. (2002). Bayesian methods for generalized linear models with covariates missing at random. *Canadian Journal of Statistics*, *30*(1), 55–78. doi: https://doi.org/10.2307/3315865

Koskinen, J. H., Robins, G. L., Wang, P., & Pattison, P. E. (2013). Bayesian analysis for partially observed network data, missing ties, attributes and actors. *Social networks*, *35*(4), 514–527. doi: https://doi.org/10.1016/j.socnet.2013.07.003

Lee, S.-Y., & Tang, N.-S. (2006). Analysis of nonlinear structural equation models with nonignorable missing covariates and ordered categorical data. *Statistica Sinica*, 1117–1141.

Little, R. J. (1988). A test of missing completely at random for multivariate data with missing values. *Journal of* *the American statistical Association*, *83*(404), 1198–1202. doi: https://doi.org/10.1080/01621459.1988.10478722

Little, R. J. (1994). A class of pattern-mixture models for normal incomplete data. *Biometrika*, *81*(3), 471–483. doi: https://doi.org/10.1093/biomet/81.3.471

Ma, Z., & Chen, G. (2018). Bayesian methods for dealing with missing data problems. *Journal of the Korean Statistical Society*, *47*(3), 297–313. doi: https://doi.org/10.1016/j.jkss.2018.03.002

Neal, R. M. (1992). Bayesian mixture modeling. In *Maximum entropy and bayesian methods* (pp. 197–211). Springer. doi: https://doi.org/10.1007/978-94-017-2219-3_14

Plummer, M., Best, N., Cowles, K., & Vines, K. (2006). Coda: convergence diagnosis and output analysis for mcmc. *R news*, *6*(1), 7–11.

Plummer, M., et al. (2003). Jags: A program for analysis of bayesian graphical models using gibbs sampling. In *Proceedings of the 3rd international workshop on distributed statistical computing* (Vol. 124, pp. 1–10).

Plummer, M., Stukalov, A., & Denwood, M. (2022). rjags: Bayesian graphical models using mcmc [Computer software manual].

R Core Team. (2021). R: A language and environment for statistical computing [Computer software manual]. Vienna, Austria. Retrieved from `https://www.R-project.org/`

Rasmussen, C. (1999). The infinite gaussian mixture model. *Advances in neural information processing systems*, *12*.

Robins, G., Pattison, P., Kalish, Y., & Lusher, D. (2007). An introduction to exponential random graph (p*) models for social networks. *Social networks*, *29*(2), 173–191. doi: https://doi.org/10.1016/j.socnet.2006.08.002

RStudio Team. (2022). Rstudio: Integrated development environment for r [Computer software manual]. Boston, MA. Retrieved from `http://www.rstudio.com/`

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, *63*(3), 581–592. doi: https://doi.org/10.1093/biomet/63.3.581

Van Buuren, S. (2018). *Flexible imputation of missing data*. CRC press.

Zhang, Z., & Wang, L. (2012). A note on the robustness of a full bayesian method for nonignorable missing data analysis. *Brazilian Journal of Probability and Statistics*, *26*(3), 244–264. doi: https://doi.org/10.1214/10-bjps132

Zhang, Z., & Wang, L. (2013). Methods for mediation analysis with missing data. *Psychometrika*, *78*(1), 154–184. doi: https://doi.org/10.1007/s11336-012-9301-5

Appendix A Complete R code

Appendix A.1 Data Preparation

```
set.seed(10)
```

```
data = read.csv("active.csv")
data = data[, c("WS_COR", "LS_COR", "LT_COR", "AGE", "EPT")]
data = na.omit(data)
data_500 <- data[sample(nrow(data),500), ]
colnames(data_500) <- c("WS", "LS", "LT", "AGE", "EPT")

# MAR
miss <- rep(NA, nrow(data_500))
for (i in 1:nrow(data_500)){
  miss[i] <- rbinom(1, 1, data_500$AGE[i]*0.01-60*0.01)
}
data_500_mar <- data_500
data_500_mar[,"EPT"][miss==1] <- NA
summary(data_500_mar)


# MNAR selection
miss2 <- rep(NA, nrow(data_500))
for (i in 1:nrow(data_500)){
  miss2[i] <- rbinom(1, 1, data_500$EPT[i]*0.01)
}
data_500_mnar <- data_500
data_500_mnar[,"EPT"][miss2==1] <- NA
summary(data_500_mnar)

# MNAR pattern-mixture
miss3 <- rep(0, nrow(data_500))
miss3[(data_500$EPT>23) & (data_500$EPT<27)] <- 1
data_500_mnar_2 <- data_500
data_500_mnar_2[,"EPT"][miss3==1] <- NA
summary(data_500_mnar_2)
```

Appendix A.2 Ignorable Missingness Model

```
library(runjags)

ignorable <- "
model{
  # likelihood
  for (i in 1:N){
    EPT[i] ~ dnorm(mu.EPT[i], pre.EPT)
    mu.EPT[i] <- b0 + b1*REASON[i] + b2*AGE[i]
    WS[i] ~ dnorm(mu1[i], pre.WS)
    LT[i] ~ dnorm(mu2[i], pre.LT)
    LS[i] ~ dnorm(mu3[i], pre.LS)
    mu1[i] <- REASON[i] + k.WS
    mu2[i] <- l.LT*REASON[i] + k.LT
    mu3[i] <- l.LS*REASON[i] + k.LS
    REASON[i] ~ dnorm(0, pre.REASON)
  }

  # priors
  # regression model
  b0 ~ dnorm(0, pre.b)
  b1 ~ dnorm(0, pre.b)
  b2 ~ dnorm(0, pre.b)
  pre.b ~ dgamma(.001,.001)
  pre.EPT ~ dgamma(0.001, 0.001)

  # factor model
  l.LT ~ dnorm(0, 0.001)
  l.LS ~ dnorm(0, 0.001)
  pre.WS ~ dgamma(0.001, 0.001)
  pre.LT ~ dgamma(0.001, 0.001)
  pre.LS ~ dgamma(0.001, 0.001)
  k.WS ~ dnorm(0, 0.001)
  k.LT ~ dnorm(0, 0.001)
  k.LS ~ dnorm(0, 0.001)
  pre.REASON ~ dgamma(0.001, 0.001)

  # variances
  var.EPT <- 1/pre.EPT
  var.WS <- 1/pre.WS
  var.LT <- 1/pre.LT
  var.LS <- 1/pre.LS
  var.REASON <- 1/pre.REASON
}

"
data <- list(N = 500,
             EPT = data_500_mar$EPT,
             AGE = data_500_mar$AGE,
             WS = data_500_mar$WS,
             LT = data_500_mar$LT,
             LS = data_500_mar$LS)

inits <- list(list(b0=0, b1=0, b2=0, l.LT=0, l.LS=0,
                   k.WS=0, k.LS=0, k.LT=0,
                   pre.WS=0.1, pre.LS=0.1, pre.LT=0.1,
                   pre.EPT=0.1, pre.REASON=0.1),
              list(b0=1, b1=1, b2=1, l.LT=1, l.LS=1,
                  k.WS=1, k.LS=1, k.LT=1,
                  pre.WS=0.2, pre.LS=0.2, pre.LT=0.2,
                  pre.EPT=0.2, pre.REASON=0.2))

out1 <- run.jags(model=ignorable,
                 monitor=c("b0", "b1", "b2",
                "l.LT", "l.LS",
                "k.WS", "k.LT", "k.LS",
                "var.WS", "var.LT",
                "var.LS", "var.EPT", "var.REASON"),
                 data=data, n.chains=2,
                 inits=inits, method="simple",
                 adapt=1000, burnin = 3000,
                 sample=1000000,
                 thin=1,
                 keep.jags.files=FALSE,
                 tempdir=TRUE)

out1
plot(out1)
```

Appendix A.3 Selection Model

```
selection<- "
model{
  # likelihood
  for (i in 1:N){
    R[i] ~ dbern(p[i])
    logit(p[i]) = a0 + a1*EPT[i]
    EPT[i] ~ dnorm(mu.EPT[i], pre.EPT)
    mu.EPT[i] <- b0 + b1*REASON[i] + b2*AGE[i]
    WS[i] ~ dnorm(mu1[i], pre.WS)
    LT[i] ~ dnorm(mu2[i], pre.LT)
    LS[i] ~ dnorm(mu3[i], pre.LS)
    mu1[i] <- REASON[i] + k.WS
    mu2[i] <- l.LT*REASON[i] + k.LT
    mu3[i] <- l.LS*REASON[i] + k.LS
    REASON[i] ~ dnorm(0, pre.REASON)
  }

  # priors
  # regression model
  b0 ~ dnorm(0, pre.b)
  b1 ~ dnorm(0, pre.b)
  b2 ~ dnorm(0, pre.b)
  a0 ~ dnorm(0, pre.a)
  a1 ~ dnorm(0, pre.a)
  pre.a ~ dgamma(.001,.001)
  pre.b ~ dgamma(.001,.001)
  pre.EPT ~ dgamma(0.001, 0.001)

  # factor model
  l.LT ~ dnorm(0, 0.001)
  l.LS ~ dnorm(0, 0.001)
  pre.WS ~ dgamma(0.001, 0.001)
  pre.LT ~ dgamma(0.001, 0.001)
  pre.LS ~ dgamma(0.001, 0.001)
  k.WS ~ dnorm(0, 0.001)
  k.LT ~ dnorm(0, 0.001)
  k.LS ~ dnorm(0, 0.001)
  pre.REASON ~ dgamma(0.001, 0.001)

  # variances
  var.EPT <- 1/pre.EPT
  var.WS <- 1/pre.WS
  var.LT <- 1/pre.LT
  var.LS <- 1/pre.LS
  var.REASON <- 1/pre.REASON
}

"
data <- list(N = 500,
             EPT = data_500_mnar$EPT,
             AGE = data_500_mnar$AGE,
             WS = data_500_mnar$WS,
             LT = data_500_mnar$LT,
             LS = data_500_mnar$LS,
             R = is.na(data_500_mnar$EPT)*1)

inits <- list(list(b0=0, b1=0, b2=0, l.LT=0, l.LS=0,
                   k.WS=0, k.LS=0, k.LT=0,
                   pre.WS=0.1, pre.LS=0.1, pre.LT=0.1,
                   pre.EPT=0.1, pre.REASON=0.1,
                   a0=0.1, a1=0.1),
              list(b0=1, b1=1, b2=1, l.LT=1, l.LS=1,
                   k.WS=1, k.LS=1, k.LT=1,
                   pre.WS=0.2, pre.LS=0.2, pre.LT=0.2,
                   pre.EPT=0.2, pre.REASON=0.2,
                   a0=0.2, a1=0.2))
```

```
out1 <- run.jags(model=selection,
                 monitor=c("b0", "b1", "b2",
                 "a0", "a1",
                 "l.LT", "l.LS", "k.WS",
                 "k.LT", "k.LS",
                 "var.WS", "var.LT",
                 "var.LS", "var.EPT", "var.REASON"),
                 data=data, n.chains=2,
                 inits=inits, method="simple",
                 adapt=1000, burnin = 3000,
                 sample=1000000,
                 thin=1,
                 keep.jags.files=FALSE,
                 tempdir=TRUE)
out1
plot(out1)
```

Appendix A.4 Pattern-Mixture Model

```
pmm <- "
model{
  # likelihood
  for (i in 1:N){
    EPT[i] ~ dnorm(mu.EPT[i], pre.EPT)
    mu.EPT[i] <- b0[R[i]+1] + b1[R[i]+1]*REASON[i] + b2*AGE[i]
    WS[i] ~ dnorm(mu1[i], pre.WS)
    LT[i] ~ dnorm(mu2[i], pre.LT)
    LS[i] ~ dnorm(mu3[i], pre.LS)
    mu1[i] <- REASON[i] + k.WS
    mu2[i] <- l.LT*REASON[i] + k.LT
    mu3[i] <- l.LS*REASON[i] + k.LS
    REASON[i] ~ dnorm(0, pre.REASON)
  }

  # priors
  # regression model
  b0[1] ~ dnorm(0, pre.b) # non-missing
  b0[2] ~ dnorm(b0[1]+05, pre.b) #missing
  b1[1] ~ dnorm(0, pre.b) # non-missing
  b1[2] ~ dnorm(b1[1]-0.5, pre.b) #missing
  b2 ~ dnorm(0, pre.b)
  pre.b ~ dgamma(.001,.001)
  pre.EPT ~ dgamma(0.001, 0.001)
  # factor model
  l.LT ~ dnorm(0, 0.001)
  l.LS ~ dnorm(0, 0.001)
  pre.WS ~ dgamma(0.001, 0.001)
  pre.LT ~ dgamma(0.001, 0.001)
  pre.LS ~ dgamma(0.001, 0.001)
  k.WS ~ dnorm(0, 0.001)
  k.LT ~ dnorm(0, 0.001)
  k.LS ~ dnorm(0, 0.001)
  pre.REASON ~ dgamma(0.001, 0.001)

  # variances
  var.EPT <- 1/pre.EPT
  var.WS <- 1/pre.WS
  var.LT <- 1/pre.LT
  var.LS <- 1/pre.LS
  var.REASON <- 1/pre.REASON
}

"

data <- list(N = 500,
             EPT = data_500_mnar_2$EPT,
             AGE = data_500_mnar_2$AGE,
             WS = data_500_mnar_2$WS,
             LT = data_500_mnar_2$LT,
             LS = data_500_mnar_2$LS,
             R = is.na(data_500_mnar_2$EPT)*1)

inits <- list(list(b0=c(0,1), b1=c(1,0), b2=0,
                   l.LT=0, l.LS=0, k.WS=0,
                   k.LS=0, k.LT=0,
                   pre.WS=0.1, pre.LS=0.1, pre.LT=0.1,
                   pre.EPT=0.1, pre.REASON=0.1),
              list(b0=c(1,2), b1=c(2,1), b2=1,
                   l.LT=1, l.LS=1, k.WS=1,
                   k.LS=1, k.LT=1,
                   pre.WS=0.2, pre.LS=0.2, pre.LT=0.2,
                   pre.EPT=0.2, pre.REASON=0.2))

out1 <- run.jags(model=pmm,
                 monitor=c("b0", "b1", "b2",
                "l.LT", "l.LS",
                "k.WS", "k.LT", "k.LS",
                "var.WS", "var.LT",
                    "var.LS", "var.EPT", "var.REASON"),
                     data=data, n.chains=2,
                     inits=inits, method="simple",
                     adapt=1000, burnin = 3000,
                     sample=1000000,
                     thin=1,
                     keep.jags.files=FALSE,
                     tempdir=TRUE)
out1
plot(out1)
```

Journal of Behavioral Data Science, 2022, *2* (2), 127–155.
DOI: `https://doi.org/10.35566/jbds/v2n2/qiu`

A Tutorial on Bayesian Latent Class Analysis Using JAGS

Meng Qiu[1]

Department of Psychology, University of Notre Dame, Notre Dame, USA
`mqiu@nd.edu`

Abstract. This tutorial introduces readers to latent class analysis (LCA) as a model-based approach to understand the unobserved heterogeneity in a population. Given the growing popularity of LCA, we aim to equip readers with theoretical fundamentals as well as computational tools. We outline some potential pitfalls of LCA and suggest related solutions. Moreover, we demonstrate how to conduct frequentist and Bayesian LCA in R with real and simulated data. To ease learning, the analysis is broken down into a series of simple steps. Beyond the simple LCA, two extensions including mixed-model LCA and growth curve LCA are provided to aid readers' transition to more advanced models. The complete R code and data set are provided.

Keywords: Latent class analysis · Mixture models · Bayesian analysis

1 Introduction

Latent class analysis (LCA) is a powerful mixture model that can be used to group individuals into homogeneous classes, types, or categories based on the responses to a set of observed variables or items. An important usage of LCA is to develop typologies based on the characteristics of the identified classes. LCA has been applied in a variety of substantive fields, such as profiles of personality (Merz & Roesch, 2011), differential diagnosis among mental disorders (Cloitre, Garvert, Weiss, Carlson, & Bryant, 2014), and dietary patterns among older adults (Harrington, Dahly, Fitzgerald, Gilthorpe, & Perry, 2014). Overviews of LCA can be found in Collins and Lanza (2010) and Depaoli (2021). Related and more complex models are discussed in Hancock, Harring, and Macready (2019).

In this tutorial, readers will learn how to perform LCA and two of its extensions using Bayesian methods. Real and simulated examples are adopted for illustration. The platform that will be used is R with the **JAGS** program installed. The reminder of the tutorial includes the following main sections. In Section 2, we provide a more formal introduction to LCA, where the LCA for binary items will be described in particular. Three fundamental issues associated with LCA are covered in Section 3. In Section 4, a real dataset used for illustration will be briefly introduced. The conventional LCA process is introduced in Section 5, which is followed by its Bayesian counterpart in Section 6. Section 7 provides readers with two related extensions. Section 8 displays a guidance for wrapping up the LCA results. The tutorial ends with a brief discussion.

2 Model and Notation

The LCA models are under the umbrella of finite mixture models (McLachlan & Peel, 2000), where observations are assumed to have arisen from one of the components, each being modeled by a density function from a parametric family (e.g., exponential). A K-component mixture density of a J-dimensional random vector $\boldsymbol{y}_i$ can be expressed as

$$f\left(\boldsymbol{y}_i;\boldsymbol{\theta}\right)=\sum_{k=1}^{K}P\left(c_i=k\right)f\left(\boldsymbol{y}_i\mid c_i=k\right)=\sum_{k=1}^{K}w_k f\left(\boldsymbol{y}_i;\boldsymbol{\lambda}_k\right),$$

where w_k indicates the mixing proportion[1] of the k-th component with $\sum_k w_k = 1$, $\boldsymbol{\lambda}_k$ the vector of unknown parameters of the k-th component, and $f\left(\boldsymbol{y}_i;\boldsymbol{\lambda}_k\right)$ the k-th component density. Also, we introduce a latent classification variable, c_i, to represent the i-th individual's class membership, where c_i takes on discrete values $1, ..., K$, so that $c_i = k$ indicates that the i-th observation belongs to the k-th class.

Although not realistic, there is one primary assumption, *local independence*, that needs to be met in the traditional LCA. This assumption implies that the items are independent of each other given latent class, meaning that latent class membership explains *all* of the shared variance among the items. The advantage of making the assumption is that it simplifies the component-level density function to a product of item-level probability densities as follows

$$f\left(\boldsymbol{y}_i\mid c_i=k\right)=\prod_{j=1}^{J}f\left(y_{ij}\mid c_i=k\right).$$

With the estimates, the posterior probability that an individual belongs to class k can be calculated using Bayes' rule

$$\hat{f}\left(c_i=k\mid\boldsymbol{y}_i\right)=\frac{\widehat{w}_k f\left(\boldsymbol{y}_i;\widehat{\boldsymbol{\theta}}_k\right)}{\sum_{v=1}^{K}\widehat{w}_v f\left(\boldsymbol{y}_i;\widehat{\boldsymbol{\theta}}_v\right)}.$$

Now, let's consider an LCA for binary items. Suppose we observe J dichotomous variables, each of which contains $\{0,1\}$ possible outcomes, for individuals $i = 1,\ldots,N$. We denote as $w_1,\ldots,w_K$ the K mixing proportions with

[1] The terms mixing proportions, weights, and class sizes are used interchangeably in this tutorial.

$\sum_{k=1}^{K} w_k = 1$. Let y_{ij} be the observed value of the j-th variable such that $y_{ij} = 1$ if individual i endorses the j-th item, and $y_{ij} = 0$ otherwise. Let π_{jk} denote the item response probability (IRP) representing how likely an individual in class k endorses the j-th item. Then the probability that an individual i in class k produces a particular set of J outcomes on the items, assuming local independence, is the product

$$f\left(\boldsymbol{y}_i \mid c_i = k\right) = \prod_{j=1}^{J} \pi_{jk}^{y_{ij}} \left(1 - \pi_{jk}\right)^{1-y_{ij}} .$$

Therefore, the overall likelihood function across the K classes for $\boldsymbol{y}_i$ is the weighted sum

$$f\left(\boldsymbol{y}_i\right) = \sum_{k=1}^{K} w_k \prod_{j=1}^{J} \pi_{jk}^{y_{ij}} \left(1 - \pi_{jk}\right)^{1-y_{ij}} ,$$

and the posterior classification probability can be written as

$$\hat{p}\left(c_i = k \mid \boldsymbol{y}_i\right) = \frac{\widehat{w}_k \prod_{j=1}^{J} \hat{\pi}_{jk}^{y_{ij}} \left(1 - \hat{\pi}_{jk}\right)^{1-y_{ij}}}{\sum_{v=1}^{K} \widehat{w}_v \prod_{j=1}^{J} \hat{\pi}_{jv}^{y_{ij}} \left(1 - \hat{\pi}_{jv}\right)^{1-y_{ij}}}.$$

3 Fundamental Issues in LCA

3.1 Local Maxima

The basic idea of parameter estimation is to find the parameter estimates that maximize the log-likelihood function; i.e., find those that yield the greatest likelihood of having generated the observed data. Therefore, we are usually looking for the global maxima. However, finding the global maximum can be particularly challenging for a gradient ascent algorithm (e.g., EM) when the log-likelihood function of a LCA model is non-concave and has multiple local maxima.

To avoid arriving at a local maximum, it is always preferable to estimate the model a couple of times with different sets of random initial values. The solution that the majority of the sets converge to can then be considered the maximum likelihood solution (McLachlan & Peel, 2000). Many software packages, such as the **poLCA** R package, have implemented the use of multiple sets of random initial values.

3.2 Boundary Parameter Estimates

It frequently occurs that one or more maximum likelihood (ML) estimates of LCA models lie on the boundary of the parameter space, i.e., the estimates are 0 or 1. This issue not only results in numerical problems in the computation of the variance-covariance matrix, but renders the provided confidence intervals and significance tests for the parameters meaningless. Nonetheless, boundary estimates can be readily avoided by imposing priors in Bayesian inference.

3.3 Label Switching

For a mixture model with K components, there are $K!$ possible permutations of the labels. Label switching refers to the phenomenon where the likelihood of a mixture model is invariant for any permutations of its component labels.

Let $\mathcal{P}_K$ be the set of $K!$ permutations of $\{1, \ldots, K\}$. If for some $\rho \in \mathcal{P}_K$, define $\boldsymbol{\theta}^\rho := (w_{\rho(1)}, \ldots, w_{\rho(K)}, \boldsymbol{\lambda}_{\rho(1)}, \ldots, \boldsymbol{\lambda}_{\rho(K)})$, then for any $\rho, \nu \in \mathcal{P}_K$,

$$\begin{aligned} p(\boldsymbol{y}_i \mid \boldsymbol{\theta}^\rho) &= \sum_{k=1}^{K} w^{\rho(k)} f\left(\boldsymbol{y}_i \mid \boldsymbol{\lambda}^{\rho(k)}\right) \\ &= \sum_{k=1}^{K} w^{\nu(k)} f\left(\boldsymbol{y}_i \mid \boldsymbol{\lambda}^{\nu(k)}\right) \\ &= p(\boldsymbol{y}_i \mid \boldsymbol{\theta}^\nu). \end{aligned}$$

In frequentist mixture models, label switching arises when working with resampling methods (e.g., bootstrap) and simulation studies. Label switching is a well-known and fundamental issue in Bayesian mixture analysis as well. In Bayesian inference, if there is no prior information that distinguishes between the mixture components, and the prior distribution is the same for the permutations of $\boldsymbol{\theta}$, then the posterior distribution will be symmetric. When such information is available, we can use a prior that imposes a constraint that makes the components unique. However, such a prior might no longer be conjugate, and the Gibbs sampler could lose its simplicity. In addition, the fact that label switching occurs both within and between Markov chains further complicates label switching.

Fortunately, ample relabeling approaches are available for correcting the label switching problem in different scenarios. Approaches proposed for frequentist inference can be found in Yao (2015) and O'Hagan et al. (2019). For Bayesian inference, Papastamoulis (2016) provides ad-hoc procedures relabeling MCMC samples.

In the following example, we demonstrate the issue in a Markov chain. Plotted are MCMC traces for the class weights of a four-class solution using a simulated dataset. In the plot, one color indicates one parameter. The left-hand panel displays the trace plot of the class weights before relabeling, where signs of label switching are presented by the iterations drifted to different classes; whereas the right-hand panel represents the trace plot after relabeling, where no drifted iterations are found. Illustrative R code for solving label switching is available in the supplementary materials.

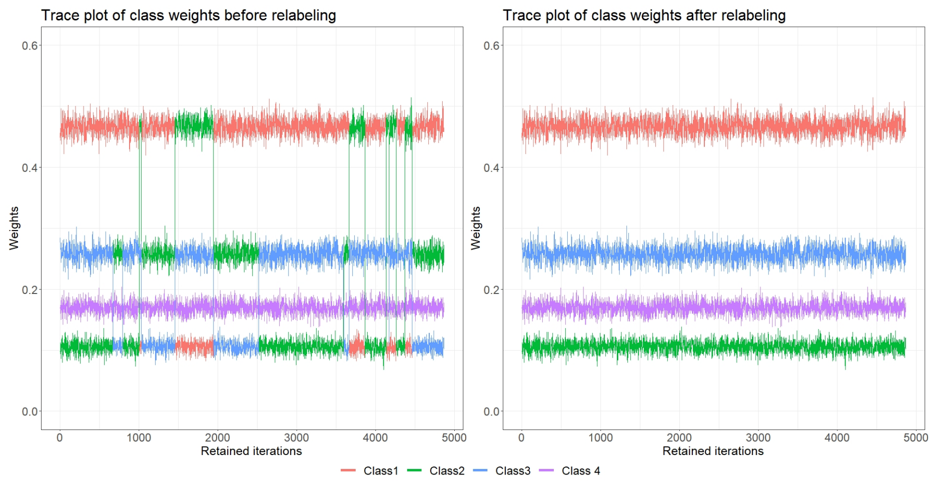


Figure 1. Illustration of the label switching issue based on a simulated dataset.

4 Real Data: National Youth Risk Behavior Survey 2019 (YRBS2019)

The National Youth Risk Behavior Survey (YRBS) is a school-based cross-sectional survey that has been conducted biennially by the Centers for Disease Control and Prevention (CDC) since 1991 to collect data on health habits and experiences among high school students across the United States.

To discover the unobserved types of health behaviors, we follow a previous study (Xiao, Romanelli, & Lindsey, 2019) and conduct an LCA on 13 health-behavior items in YRBS2019 that encompass four domains including: 1) diet (consumption of breakfast, fruits, juices, vegetables, milk, water, and soda), 2) physical activities (moderate and vigorous physical activity [MVPA], muscle-strengthening exercise [MSEs], and sports team participation [STP]), 3) sleeping time, and 4) media use (television [TV], computer/video games).

The items are dichotomized to 0 and 1, in which three items are reverse coded including television, computer/video games, and consumption of soda. A sample of 1,000 complete cases will be used for the analysis.

5 The Conventional LCA

5.1 Class Enumeration

Determining the number of classes is always a challenging task in practice. The decision of how many classes to retain in LCA is conventionally made through the so-called *class enumeration* process. Class enumeration includes fitting a series of LCA models with increment of one in K and selecting the “best” model via certain criterion (e.g., information criteria) or a set of criteria. To compare the criteria across the fittings, we often tabulate or plot the fit information for each model fitting, and study patterns to pick the optimal K. This process will be illustrated in Section 5.4. For a more detailed introduction to model fit and model selection in LCA, interested readers can refer to Section 4.3 of Collins and Lanza (2010).

5.2 Classification Uncertainty

A popular approach to summarizing uncertainty in posterior classification is entropy. In LCA, the entropy can be expressed as

$$EN(\boldsymbol{\alpha}) = -\sum_{i=1}^{n}\sum_{k=1}^{K} \alpha_{ik} \log \alpha_{ik},$$

where α_{ik} represents the posterior classification probability of individual i defined as

$$\hat{\alpha}_{ik} = \frac{\hat{w}_k f_k\left(\boldsymbol{y}_i; \hat{\boldsymbol{\theta}}_k\right)}{\sum_{v=1}^{K} \hat{w}_v f_v\left(\boldsymbol{y}_i; \hat{\boldsymbol{\theta}}_v\right)}.$$

A normalized version of EN, called relative entropy (RE), that scales EN to the interval $[0, 1]$ has been commonly used in LCA and is defined as

$$RE = 1 - EN/(N \log K),$$

with RE closer to 1 indicating less classification uncertainty and clearer assignment of individuals to latent classes. A RE greater than 0.6 is generally considered to be satisfactory class separation (Asparouhov & Muthen, 2014).

5.3 Interpretation of Latent Classes

The last step of an LCA analysis is the interpretation of the retained solution, which involves more human judgement. The researchers need to examine the class-specific parameter estimates and then label each of the individual classes. Such labels should be related to the included items. It is beneficial to display the estimates in a plot (e.g., line plot). However, if a large number of classes is identified, it is more sensible to number the classes instead of assigning labels that cannot be easily distinguished verbally anymore. Another issue that may merit consideration is when the emergent classes differ merely quantitatively. In other words, the plotted lines are generally parallel. This phenomenon could indicate that a single sample was coerced into K levels, such as low, medium, and high severity. When confronted with such classes, the solution should be interpreted cautiously.

5.4 Software

LCA can be performed using a variety of commercial statistical packages, including **M*plus*** (Muthen & Muthen, 1998-2017), **SAS** (SAS, 2016), **STATA** (STATA, 1985-2019), and **Latent GOLD** (Vermunt & Magidson, 2016). These commercial packages are user-friendly, can handle a wide range of mixture models, and offer cutting-edge approaches to handling missing data, covariates, and distal outcomes.

Various free R packages pertinent to mixture modeling are listed on the *Cran Task Views: Psychometric Models and Methods and Clusters* (`https://cran.r-project.org/web/views`), such as **poLCA** (Linzer & Lewis, 2011), **MCLUST** (Scrucca, Fop, Murphy, & Raftery, 2016), and **tidyLPA** (Rosenberg, Beymer, Anderson, van Lissa, & Schmidt, 2018). Haughton et al. (2009) provides a review of three packages, namely, **Latent GOLD**, **MCLUST**, and **poLCA**.

In this tutorial, we demonstrate the implementation of the frequentist LCA with **poLCA**, and the Bayesian LCA with **JAGS**.

5.5 Demonstration of LCA on YRBS2019 Using poLCA Package

The following chunk of code loads required packages for the entire tutorial.

```
library("poLCA")           # To use poLCA function
library("label.switching") # To address label switching
library("runjags")         # To use JAGS via R
library("glue")            # To insert R code within strings
library("gt")              # For pretty tables
library("knitr")           # To use the "kable" function
library("kableExtra")      # For more about kable
library("cowplot")         # For pretty plots
library("scatterplot3d")   # For 3-D plot
library("ggpubr")          # To use the "ggarrange" function
library("coda")            # To use Geweke's diagnostic test
library("MASS")            # To use the "mvrnorm" function
library("tidyverse")       # For everything else...
```

This chunk of code loads saved datasets and outputs for the following sections to reduce knitting time.

```
load("C:/Users/Chris/Desktop/LCA/objects.RData")
```

Step 1: Conduct class enumeration To use the `poLCA()` function, the items must be coded as integer values starting at one for the first outcome category, and increasing to the maximum number of outcomes for each variable. Consequently, we add one to all the responses resulting in binary outcomes of $\{1, 2\}$.

```
dat <- dat.yrbs + 1 # poLCA only allows positive integers
```

Notice that a specific R function `lca_re()` is created for computing relative entropy. The reason is that **poLCA** defines entropy as a measure of dispersion (or concentration) in a probability mass function, which is different from the widely used definition (e.g., in ***Mplus***). For computational details, readers can refer to the help document of `poLCA.entropy()` function.

```
# Function for computing relative entropy
lca_re <- function(x) {
  nom <- (sum(-x$posterior*log(x$posterior)))
  denom <- (nrow(x$posterior)*log(ncol(x$posterior)))
  re <- 1 - (nom/denom)
  if (is.nan(re) == TRUE) re <- NA
  return(re)
}
```

To specify an LCA model, `poLCA()` requires users to provide a model formula. For the basic LCA model without covariates, the formula takes the following form:

```
f <- cbind(Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11, Y12, Y13)
             ~ 1
```

where the observed variables or items are bound together within `cbind(Y1, Y2, Y3, ...)`, and the 1 indicates the LCA model without covariates. For LCA with covariates, one must substitute the 1 with a function of covariates as follows:

```
f <- cbind(Y1, Y2, Y3, Y4, Y5, Y6, Y7, Y8, Y9, Y10, Y11, Y12, Y13)
             ~ X1 + X2 + X3
```

When there are a large amount of items, as in our example, we can define the formula with the following code to save line space and typing time:

```
J <- ncol(dat) # number of items
f <- as.formula(paste("cbind(",paste(paste0("Y",1:J),collapse=","),
")","~1"))
# This is equivalent to: f <- cbind(Y1,Y2,Y3,Y4,Y5,Y6,Y7,Y8,Y9,Y10,
#                                     Y11,Y12,Y13) ~ 1
```

To estimate the desired LCA model via the `poLCA()` function, users need to pass the model formula, the dataset stored as a data frame, and the number of classes to the `formula`, `data`, and `nclass` arguments, respectively. Information about the remaining optional arguments of the function can be obtained by entering the command `help(poLCA)` or simply `?poLCA` at the R console window. Nevertheless, we would like to place an emphasis on the `nrep` argument, which specifies the number of times to estimate the model using different sets of random starting values. As discussed in Section 3.1, this is desirable to prevent solutions from converging to local instead of global maxima of the likelihood function. For example, we set `nrep` to 20 in the following code. Complex models often require more replications, especially in high-stakes applications.

In addition, it should be noted that `poLCA()` only provides two fit indices, AIC and BIC, by default. Other fit indices (e.g., aBIC) can be calculated manually based on the model results as shown in the code below.

```
# Class enumeration from K=1 to K=6
out_lca <- list()   # container of model fittings
npar <- ll <- bic <- abic <- caic <- awe <- re <- c() # containers
        # of other information
set.seed(123)
for(k in 1:6){
  fit <- poLCA(formula=f, data=dat, nclass=k, maxiter=10000,
               tol=1e-5, nrep=20, verbose=F, calc.se=T)
  out_lca[[k]] <- fit
  npar[k] <- fit$npar
  ll[k]   <- fit$llik
  bic[k]  <- fit$bic
  abic[k] <- -2*(fit$llik) + fit$npar*log((fit$Nobs+2)/24)
  caic[k] <- -2*(fit$llik) + fit$npar*(log(fit$Nobs)+1)
  awe[k]  <- -2*(fit$llik) + 2*(fit$npar)*(log(fit$Nobs)+1.5)
  re[k]   <- round(lca_re(fit), 3)
}
class <- paste0("Class-", 1:6)

# Store information in a data frame
poLCA.tab <- data.frame("Class"=class, "Npar"=npar, "LL"=ll,
    "BIC"=bic, "aBIC"=abic, "CAIC"=caic, "AWE"=awe, "RE"=re)
```

Step 2: Model fit summary table See Table 1 for the summary of fit.

Step 3: Elbow plot of information criteria It is useful to plot the values of the selected information criteria (ICs) for visual inspection. Lower IC values signify a more optimal balance of model fit and parsimony. Ideally, a minimum value in the set of fittings indicates the optimal solution. However, it is not uncommon in practice that ICs continue to decrease as K increases. That is, there is no global minimum. In such instances, the K at an "elbow" of point of "diminishing returns" in model fit indices should be selected as the best solution. For the empirical example, the following elbow plot in Figure 2 suggests that the 2-class solution fits best based on AWE, whereas the 4-class solution is supported by CAIC, BIC, and aBIC. Therefore, the 4-class solution appears to be an appealing candidate. Next, this solution's relative entropy is 0.748 (see Table 1), which is relatively high (> 0.6) and indicative of satisfactory classification quality. Considering the current information, we decide on the 4-class solution.

Table 1. Model fit summary table based on poLCA outputs.

Class	Npar	LL	BIC	aBIC	CAIC	AWE	RE
Class-1	13	-7575.656	15241.11	15199.82	15254.11	15369.91	NA
Class-2	27	-7278.148	14742.81	14657.05	14769.81	15010.31	0.666
Class-3	41	-7171.830	14626.88	14496.66	14667.88	15033.10	0.721
Class-4	55	-7103.205	14586.34	14411.65	14641.34	15131.26	0.748
Class-5	69	-7075.080	14626.79	14407.65	14695.79	15310.43	0.764
Class-6	83	-7050.491	14674.33	14410.71	14757.33	15496.67	0.704

[1] Npar = number of parameters;

[2] LL = log data likelihood;

[3] BIC = bayesian information criterion;

[4] aBIC = sample size adjusted BIC;

[5] CAIC = consistent Akaike information criterion;

[6] AWE = approximate weight of evidence criterion;

[7] RE = relative entropy.

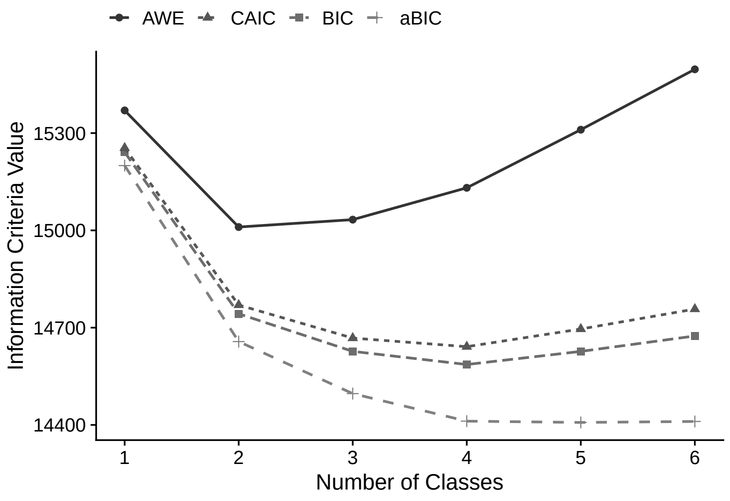


Figure 2. Elbow plot of the information criteria for the one-to six-group LCA of health behaviors.

Step 4: Plot class profiles The profiles of the latent classes are shown in Figure 3.

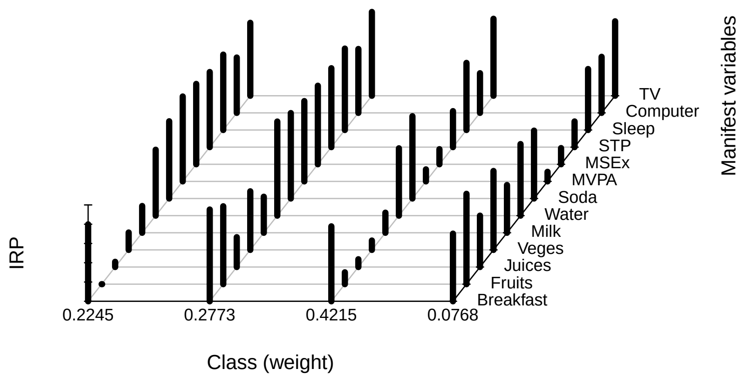


Figure 3. Conditional probability of endorsing health-behavior items for each class.

Step 5: Interpretation of emergent classes According to the above item response probability plot, we can assign labels to the emergent classes. The first class is of medium size (22.44%), and shows low engagement in health diet behavior, high engagement in exercise, and moderately high engagement in computer use. Hence, it can be characterized as the class of *irregular diet, high exercise, moderately high computer use.* The second class is small (7.67%), and characterized by the highest engagement in healthy diet behaviors, low engagement in exercise, and relatively low computer use. We call this class the *healthy diet, low exercise, relatively low computer use.* The third class is the largest (42.16%), and can be called the *lowest engagement in health-promoting behaviors* due to the low engagement in healthy diet behaviors and physical activities and the highest engagement in computer use. The fourth class, *consistent engagement in health-promoting behaviors*, is of medium size (27.73%) and demonstrates moderate probabilities of a healthy dietary pattern, frequently physical activities, high probability of longer sleeping time, and low probability of playing computer more than 3 hours per day.

6 Bayesian LCA Using JAGS

6.1 Specification of Priors

The model parameters of particular interest in LCA with binary items include mixing proportions $\boldsymbol{w}$ and binary IRPs $\boldsymbol{\pi}$. Here, $\boldsymbol{w}$ is assumed to follow a Dirichlet distribution denoted as

$$\boldsymbol{w} \sim \mathcal{D}\left(d_1, \ldots, d_K\right)$$

with the hyperparameters $d_1, \ldots, d_K$ which represent the prior proportion of individuals in each of the K classes. The conjugate prior for IRP is the Beta distribution denoted as

$$\pi_{jk} \sim \mathcal{B}\left(\alpha, \beta\right)$$

where the hyperparameters α and β represents the prior sample sizes for the number of individuals answering "Yes" and "No", respectively.

6.2 Convergence Diagnostic

The convergence diagnostic method adopted in this tutorial is Geweke's test, which compares the location of the sampled parameter on two different time intervals of the chain. If the mean values of the parameter in the two time intervals are close to each other we then assume that the two parts of the chain have similar locations in the state space, and it is assumed that the two parts come from the same distribution. An absolute value of the z-score produced by Geweke's test above 2 could be considered as potentially problematic.

6.3 Demonstration of Bayesian LCA on YRBS2019

Step 1: Define JAGS model The following code specifies a **JAGS** model and stores the model in an R object called `lca_bin`. In fact, the model can be specified as a character string within R, or in an external text file. The former avoids the need for multiple text files, whereas the latter is preferred for more complex model formulations. In the tutorial, we adopt the former way.

Every model specification must begin with informing **JAGS** that it is a model specification using the `model{}` block. Within the model block, we specify data likelihood for every single data point using a `for` loop. Note that, there are two types of data in a mixture model−unobserved class membership $\boldsymbol{z}$ and observed data $\boldsymbol{y}$. Thus, we specify likelihood for $\boldsymbol{z}$ and $\boldsymbol{y}$, respectively, as shown in the "likelihood specification" chunk. First, we let $z[i]$ follows a categorical distribution with the `dcat()` call where $w[1 : K]$ is a vector of non-negative mixing proportions of length K. Second, the likelihood for $\boldsymbol{y}$ is specified through a nested `for` loop because there are two indices (i.e., $i = 1, \ldots, N$ and $j = 1, \ldots, J$) associated with each data point of $\boldsymbol{y}$, and we let $y[i, j]$ follow a Bernoulli distribution with the `dbern()` call where $\pi[j, z[i]]$ represents the IRP for the j-th item within the $z[i]$-th class.

After specifying the data likelihood, we then impose priors on the model parameters (see the "prior specification" chunk). As discussed in Section 6.1, we impose Dirichlet and beta priors on mixing proportions ($\boldsymbol{w}$) and IRPs ($\boldsymbol{\pi}$), respectively.

```
# Build model: LCA for binary items
lca_bin <- "
model{
      #===========================
      # likelihood specification
      #===========================

      for(i in 1:N) {
        Z[i] ~ dcat(w[1:K]) # class membership for the i-th subject
        for(j in 1:J) {
          Y[i,j] ~ dbern(pi[j,Z[i]]) # Bernoulli density function
        }
      }

      #===========================
      # prior specification
      #===========================

      # --------- w ---------
      w[1:K] ~ ddirch(alpha[1:K])   # Dirichlet prior for
                                    # mixing proportions
      for(k in 1:K) {alpha[k] <- 1}

      # --------- pi ---------
      for(k in 1:K) {
        for(j in 1:J) {
          pi[j,k] ~ dbeta(3,3)      # beta prior for conditional
                                    # probabilities
        }
      }
}"
```

Step 2: Specify initial values Users of **JAGS** have the option of supplying their own initial values or just using those generated by the random number generators (RNGs). For user-specified initial values, list starting values for each of the parameters in the model as follows:

```
# User-specified initial values
inits <- list(list(w=rep(0.25,4), pi=matrix(rbeta(J*K.est,3,3),
              nrow=J, ncol=K.est)))
```

There are four RNGs provided by the base module in **JAGS** with the following names:

– "base::Wichmann-Hill"
– "base::Marsaglia-Multicarry"
– "base::Super-Duper"
– "base::Mersenne-Twister"

To set the starting state of the RNG, one can simply supply the name of the RNG and its seed (e.g., 111) as shown in the following code:

```
# Automatically generated initial values
inits <- list(".RNG.name"="base::Wichmann-Hill", ".RNG.seed"=111)
```

Step 3: Fit the model via JAGS We need to bundle the data and constants used in the **JAGS** model into a list that **JAGS** can read (see the `Dat` object in the following code). The call to `run.jags()` reads, complies, executes, and returns the model information along with MCMC samples and summary statistics. Before a model can be executed, the `run.jags()` function requires a valid **JAGS** model to be passed to the `model` argument and a character string of monitored parameters to the `monitor` argument.

Furthermore, the function allows users to specify additional arguments, including but not limited to: 1) the method with which to call **JAGS** (e.g., `rjags`); 2) number of Markov chains to run (e.g., `n.chain=1`); 3) the number of adaptive iterations used at the start of the chain (e.g., `adapt=1000`); 4) the number of burnin iterations (e.g., `burnin=1000`); 5) number of iterations per chain (e.g., `sample=10000`) in addition to the adaptive and burnin iterations; and 6) thinning interval for monitors (e.g., `thin=1`).

```
# Bundle data for JAGS
Dat <- list("Y"=as.matrix(dat.yrbs), "N"=N, "J"=J, "K"=4)
# Run the analysis
set.seed(1234)
out_bin <- run.jags(model=lca_bin, monitor=c('w','pi'),
                    data=Dat, inits=inits, method="rjags",
                    n.chains=1, adapt=1000, burnin=1000,
                    sample=10000, thin=1)
```

Step 4: Inspect label switching The trace plots of the weight of each class are displayed in Figure 4 and from it, no signs of label switching are found.

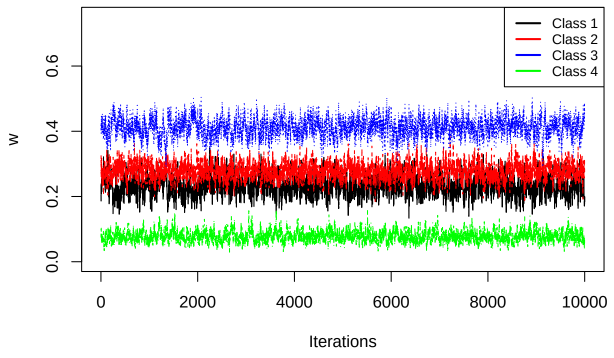


Figure 4. Trace plot of class weights for inspecting label switching.

Step 5: Adaption and burn-in Note that the `start` argument of the `window()` function include the adaption (i.e., `na`) and burn-in (i.e., `nb`) phases, so if one has 12,000 iterations with 1,000 adaption and 1,000 burn-in samples, the following code will retain the iterations from 2,001 to 12,000.

```
na <- 1000; nb <- 1000
out.burn <- window(out_bin$mcmc, start=na+nb+1)
```

Step 6: Convergence diagnostic via Geweke's test

```
con.diag <- geweke.diag(out.burn)
# |z| > 2 is potentially problematic
flag.cov <- which(abs(con.diag[[1]]$z)>2)
# double check via trace plot
traceplot(out.burn[[1]][ ,flag.cov])
```

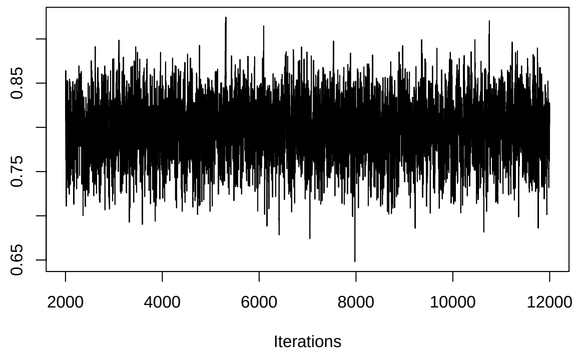


Figure 5. Trace plot of the parameter flagged by Geweke's test.

Step 7: Summarize posterior MCMC samples

```
sum.stats <- summary(out.burn)         # summary statistics
post.means <- sum.stats$statistics[,1] # posterior means

# Note: pi[j,k] is the IRP of the j-th item in the k-th class
round(post.means, digits=3)

##     w[1]     w[2]     w[3]     w[4]  pi[1,1]  pi[2,1]  pi[3,1]  pi[4,1]
##    0.234    0.277    0.411    0.078    0.792    0.047    0.069    0.179
##  pi[5,1]  pi[6,1]  pi[7,1]  pi[8,1]  pi[9,1] pi[10,1] pi[11,1] pi[12,1]
##    0.277    0.680    0.797    0.847    0.816    0.760    0.770    0.568
## pi[13,1]  pi[1,2]  pi[2,2]  pi[3,2]  pi[4,2]  pi[5,2]  pi[6,2]  pi[7,2]
##    0.747    0.943    0.783    0.310    0.613    0.378    0.963    0.881
##  pi[8,2]  pi[9,2] pi[10,2] pi[11,2] pi[12,2] pi[13,2]  pi[1,3]  pi[2,3]
##    0.817    0.799    0.815    0.842    0.663    0.867    0.773    0.136
##  pi[3,3]  pi[4,3]  pi[5,3]  pi[6,3]  pi[7,3]  pi[8,3]  pi[9,3] pi[10,3]
##    0.083    0.107    0.211    0.697    0.851    0.127    0.153    0.370
## pi[11,3] pi[12,3] pi[13,3]  pi[1,4]  pi[2,4]  pi[3,4]  pi[4,4]  pi[5,4]
##    0.696    0.415    0.798    0.677    0.851    0.562    0.770    0.512
##  pi[6,4]  pi[7,4]  pi[8,4]  pi[9,4] pi[10,4] pi[11,4] pi[12,4] pi[13,4]
##    0.723    0.663    0.160    0.220    0.289    0.598    0.553    0.730
```

Step 8: Plot class profiles for interpretation Compared with Figure 3, the class profiles yielded by the Bayesian LCA (Figure 6) look quite similar to that provided by the conventional LCA.

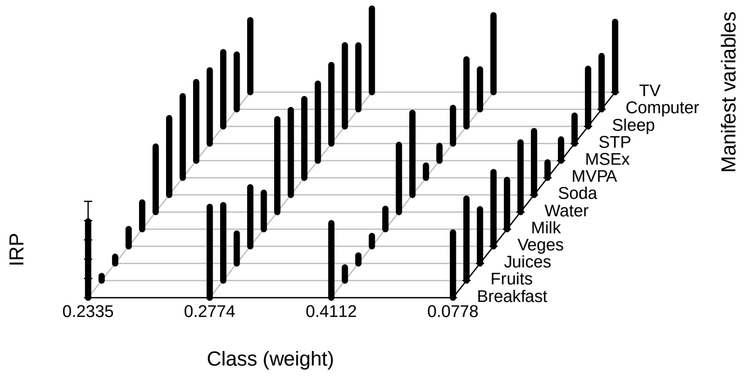


Figure 6. Probability of endorsing health-behavior items for each class.

7 Extensions

7.1 Bayesian Mixed-mode Latent Class Analysis (MMLCA)

Specification of Model This section focuses on MMLCA for which observed variables are a mix of binary and continuous data. Suppose we observe M variables consisting of J binary variables and L continuous variables for individuals $i = 1, \cdots, N$. Let u_{ij} be the observed value of the j-th binary item such that $u_{ij} = 1$ if the i-th individual endorses the item (i.e., answer "Yes"), and $u_{ij} = 0$ otherwise, where $j = 1, \ldots, J$; let v_{il} be the observed value of the l-th continuous item.

Assuming local independence, the probability that an individual i in class k produces a particular set of $M = J+L$ responses $y_i = \{u_{i1}, u_{i2}, \ldots, u_{iJ}; v_{i1}, v_{i2}, \ldots, v_{iL}\}$ is a finite mixture of conditional densities

$$f(\boldsymbol{y}_i) = \sum_{k=1}^{K} p(c_i = k) f(\boldsymbol{y}_i \mid c_i = k) = \sum_{k=1}^{K} w_k g(\boldsymbol{u}_i \mid k) g(\boldsymbol{v}_i \mid k),$$

where $g(\boldsymbol{u}_i \mid k)$ is the conditional density of the vector of observed binary items and $g(\boldsymbol{v}_i \mid k)$ is the conditional density of the vector of observed continuous items for the i-th individual given the k-th class, respectively.

Because of the local independence assumption, the two conditional densities can be further expanded. The $g(\boldsymbol{u}_i \mid k)$ can be expressed as a product of Bernoullis

$$g(\boldsymbol{u}_i \mid k) = \prod_{j=1}^{J} \pi_{jk}^{u_{ij}} (1 - \pi_{jk})^{1-u_{ij}}$$

where π_{jk} represents the class-specific item response probability (IRP) that observations in the k-th class endorse the j-th binary variable; the $g(\boldsymbol{v}_i \mid k)$ can be written as a product of univariate Gaussians

$$g(\boldsymbol{v}_i \mid k) = \prod_{l=1}^{L} \frac{1}{\sqrt{2\pi\sigma_{lk}^2}} \exp\left\{-\frac{1}{2\sigma_{lk}^2}(v_{il} - \mu_{lk})^2\right\},$$

where μ_{lk} and σ_{lk} represent the mean and standard deviation of the l-th continuous variable, respectively.

Specification of Priors Following the priors in LCA for binary items, we specify Dirichlet prior and Beta prior to mixing proportions and IRPs, respectively

$$\begin{aligned} \boldsymbol{w} &\sim \mathcal{D}(d_1, \ldots, d_K), \\ \pi_{jk} &\sim \mathcal{B}(\alpha, \beta). \end{aligned}$$

For the parameters associated with the continuous items, we specify normal prior and Gamma prior to mean and precision ($\tau = 1/\sigma^2$), respectively

$$\begin{aligned} \mu_{lk} &\sim \mathcal{N}(\mu_0, \sigma_0^2), \\ \tau_{jk} &\sim \mathcal{G}(\lambda, \kappa). \end{aligned}$$

7.2 Demonstration of Bayesian MMLCA on Simulated Data

We simulate a dataset of 500 subjects from an MMLCA model with 3 classes and 8 items (4 binary and 4 continuous). The parameters shown in Table 2 serve as the population values for the data generating model. Each subject's class membership is generated from unequal class sizes ($\boldsymbol{w}$) of 0.5, 0.3, and 0.2. For the binary items, the class-specific response probabilities ($\boldsymbol{\pi}$) are set to 0.9 in Class 1, to 0.9 for the first half of the items and 0.1 to the other half in Class 2, and to 0.1 in Class 3. For the continuous items, the class-specific item means ($\boldsymbol{\mu}$) are set to 1 in class 1, to 0 in Class 2, and to -1 in Class 3, with the item variances ($\boldsymbol{\sigma}^2$) fixed at 1 over the three classes.

Table 2. Population values for generating the dataset for MMLCA.

Class	w	π	μ	σ^2
1	0.5	0.9, 0.9, 0.9, 0.9	1, 1, 1, 1	1, 1, 1, 1
2	0.3	0.9, 0.9, 0.1, 0.1	0, 0, 0, 0	1, 1, 1, 1
3	0.2	0.1, 0.1, 0.1, 0.1	-1, -1, -1, -1	1, 1, 1, 1

Step 1: Define JAGS model

```
# Build model: LCA for binary and continuous items
lca_mix <- "
model{
      #===========================
      # likelihood specification
      #===========================

      for(i in 1:N) {
        Z[i] ~ dcat(w[1:K]) # Class membership for the i-th subject
        for(j in 1:J) {
          Y[i,j] ~ dbern(pi[j,Z[i]]) # Bernoulli density function
        }
        for(l in 1:L) {
          X[i,l] ~ dnorm(mu[l,Z[i]], tau[l,Z[i]]) # normal density
        }
      }

      #===========================
      # prior specification
      #===========================

      # --------- w ---------
      w[1:K] ~ ddirch(alpha[1:K])      # Dirichlet prior for
                                       #mixing proportions
      for(k in 1:K) {alpha[k] <- 1}

      # --------- pi ---------
      for(k in 1:K) {
        for(j in 1:J) {
          pi[j,k] ~ dbeta(3,3)         # beta prior for IRPs
        }
      }

      # --------- mu & tau ---------
      for(k in 1:K) {
        for(l in 1:L) {
          mu[l,k] ~ dnorm(0,1.0E-6)    # normal prior for mean
          tau[l,k] ~ dgamma(0.01,0.01) # gamma prior for precision
        }
      }
}"
```

Step 2: Specify initial values

```
# Automatically generated initial values
inits <- list(list(".RNG.name"="base::Wichmann-Hill",
        ".RNG.seed"=111))
```

Step 3: Fit the model via JAGS

```
# Bundle data for JAGS
Dat <- list("Y"=dat.mmlca[,1:J], "X"=dat.mmlca[,(J+1):(J+L)],
            "N"=N, "J"=J, "L"=L, "K"=3)

# Run the analysis
set.seed(1234)
out_mix <- run.jags(model=lca_mix, monitor=c('w','pi','mu','tau'),
              data=Dat, inits=inits, method="rjags", n.chains=1,
              adapt=1000, burnin=1000, sample=10000, thin=1)
```

Step 4: Inspect label switching Similarly, no signs of label switching are found in Figure 7.

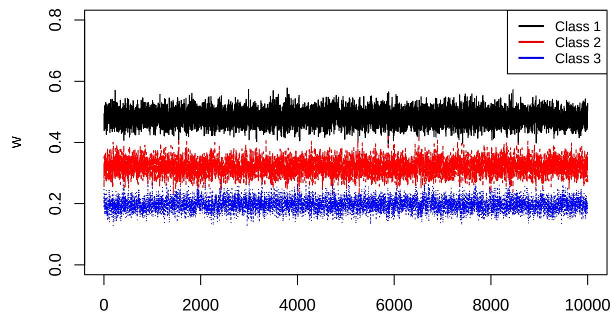


Figure 7. Trace plot of class weights for inspecting label switching.

Step 5: Adaption and burn-in

```
na <- 1000; nb <- 1000
out.burn <- window(out_mix$mcmc, start=na+nb+1)
```

Step 6: Convergence diagnostic using Geweke's test

```
con.diag <- geweke.diag(out.burn)
flag.noncov <- which(abs(con.diag[[1]]$z)>2)
```

Step 7: Summarize posterior MCMC samples

```
sum.stats <- summary(out.burn)         # summary statistics
post.means <- sum.stats$statistics[,1] # posterior means
round(post.means, digits=3)
```

```
##     w[1]     w[2]     w[3]  pi[1,1]  pi[2,1]  pi[3,1]  pi[4,1]  pi[1,2]
##    0.483    0.320    0.197    0.889    0.905    0.912    0.905    0.930
##  pi[2,2]  pi[3,2]  pi[4,2]  pi[1,3]  pi[2,3]  pi[3,3]  pi[4,3]  mu[1,1]
##    0.835    0.109    0.133    0.138    0.203    0.152    0.178    0.960
##  mu[2,1]  mu[3,1]  mu[4,1]  mu[1,2]  mu[2,2]  mu[3,2]  mu[4,2]  mu[1,3]
##    0.912    1.051    1.130    0.081    0.119    0.042   -0.043   -1.044
##  mu[2,3]  mu[3,3]  mu[4,3] tau[1,1] tau[2,1] tau[3,1] tau[4,1] tau[1,2]
##   -0.865   -0.806   -0.907    1.179    0.849    1.002    1.087    1.062
## tau[2,2] tau[3,2] tau[4,2] tau[1,3] tau[2,3] tau[3,3] tau[4,3]
##    1.100    0.893    1.162    1.061    0.942    0.957    0.706
```

Step 8: Plot class profiles for interpretation The profiles of the classes are shown in Figure 8.

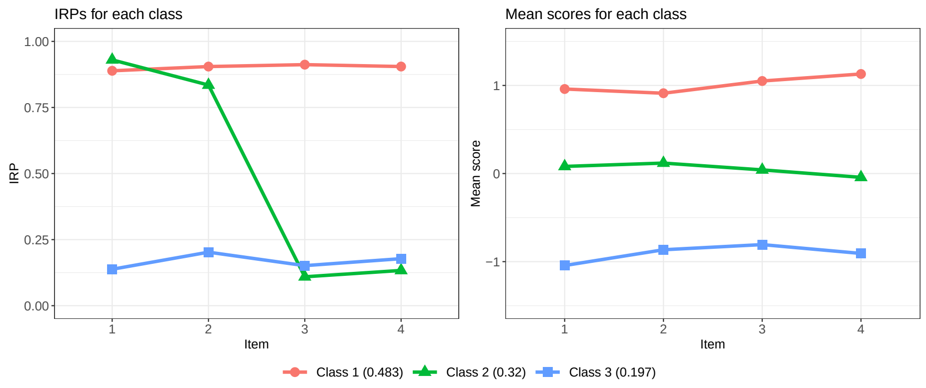


Figure 8. Probability of binary items (left panel) and mean score of continuous items (right panel) for each class.

7.3 Bayesian Latent Growth Mixture Model (LGMM)

Specification of Model The latent growth curve model (LGCM) characterizes changes in responses over time and estimate inter-individual variability in those changes. The LGCM can be decomposed into two components: the measurement model and the structural model. Suppose that individuals $i = 1, \ldots, N$ are assessed repeatedly at T time points. The measurement model can be expressed as

$$\boldsymbol{y}_i = \boldsymbol{\Lambda}\boldsymbol{b}_i + \boldsymbol{\epsilon}_i,$$

where $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{iJ})^T$ is a $T \times 1$ vector of repeated-measures for individual i, $\boldsymbol{\Lambda}$ is a matrix of factor loadings with T rows and m (number of latent factors) columns, and $\boldsymbol{\epsilon}_i$ is a $T \times 1$ vector of measurement errors. The entries of $\boldsymbol{\Lambda}$ define the shape of growth trajectories, for instance,

$$\boldsymbol{\Lambda} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \end{bmatrix}$$

represents a linear growth curve. The structural model is defined as follows

$$\boldsymbol{b}_i = \boldsymbol{\beta} + \boldsymbol{u}_i,$$

where $\boldsymbol{b}_i$ is a $m \times 1$ vector of latent factors, $\boldsymbol{\beta}$ is a $m \times 1$ vector of latent factor means, and $\boldsymbol{u}_i$ is a $m \times 1$ vector of random effects that are independent of the measurement errors. Conventional LGCM assumes that

$$\begin{aligned} \boldsymbol{\epsilon}_i &\sim N(\boldsymbol{0}, \boldsymbol{\Omega}), \\ \boldsymbol{u}_i &\sim N(\boldsymbol{0}, \boldsymbol{\Psi}), \end{aligned}$$

where $\boldsymbol{\Omega}$ is a $T \times T$ covariance matrix of measurement errors, and $\boldsymbol{\Psi}$ is a $m \times m$ covariance matrix of latent factors. In this tutorial, we follow the traditional assumption that $\boldsymbol{\Omega} = \sigma^2 \boldsymbol{I}$. Putting the two models together, $\boldsymbol{y}_i$ has the following density function

$$f(\boldsymbol{y}_i \mid \boldsymbol{\Theta}) = \Phi\left(\boldsymbol{\mu} = \boldsymbol{\Lambda}\boldsymbol{\beta},\ \boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Psi}\boldsymbol{\Lambda}^T + \boldsymbol{\Omega}\right),$$

where Θ and Ψ represent the set of all parameters and the T-dimensional multivariate normal density function, respectively.

The LGMM is formulated much the same way as the LCA in Section 2. The difference is that we now substitute the component density in LCA for binary items (e.g., product of Bernoullis) with the density of LGCM. That is, in a LGMM, each latent class describes a distinct growth trajectory. Therefore, the mean and covariance matrix can be written at the latent class level

$$\begin{aligned} \boldsymbol{\mu}_k &= \boldsymbol{\Lambda}\boldsymbol{\beta}_k, \\ \boldsymbol{\Sigma}_k &= \boldsymbol{\Lambda}\boldsymbol{\Psi}_k\boldsymbol{\Lambda}^T + \boldsymbol{\Omega}_k. \end{aligned}$$

Specification of Priors

$$\begin{aligned} \boldsymbol{w} &\sim \mathcal{D}(d_1, \ldots, d_K), \\ \beta &\sim \mathcal{N}\left(\mu_0, \sigma_0^2\right), \\ \boldsymbol{\Psi}^{-1} &\sim \mathcal{W}(\boldsymbol{V}, m), \\ 1/\sigma^2 &\sim \mathcal{G}(\alpha, \beta), \end{aligned}$$

where, for the Wishart prior, $\boldsymbol{V}$ and m represent the scale matrix and the degree of freedom, respectively.

7.4 Demonstration of Bayesian LGMM via Simulated Data

We simulate a dataset of 200 subjects from a LGMM model with 3 classes and 4 time points. The parameters shown in Table 3 serve as the population values for the data generating model. Each subject's class membership is generated from unequal class sizes ($\boldsymbol{w}$) of 0.5, 0.3, and 0.2. Growth factor means are set to $(\beta_{I1}, \beta_{S1})^T = (2, 0)^T$ in Class 1, to $(\beta_{I2}, \beta_{S2})^T = (4, -0.3)^T$ in Class 2, and to $(\beta_{I3}, \beta_{S3})^T = (6, 0.3)^T$ in Class 3. In addition, $\boldsymbol{\Psi} = \begin{pmatrix} 0.3 & 0 \\ 0 & 0.1 \end{pmatrix}$ and $\sigma^2 = 0.5$ for the three classes.

Table 3. Population values for generating the dataset for LGMM.

Class	w	β_I	β_S	Ψ	σ^2
1	0.5	2	0	0.3, 0, 0, 0.1	0.5
2	0.3	4	-0.3	0.3, 0, 0, 0.1	0.5
3	0.2	6	0.3	0.3, 0, 0, 0.1	0.5

Step 1: Define JAGS model

```
# Build model: latent growth mixture model
gmm <- "
model {
      #===========================
      # likelihood specification
      #===========================

      for (i in 1:N)  {
        Z[i] ~ dcat(w[1:K])
        for(t in 1:Time) {
          # model the growth curve
          y[i,t] ~ dnorm(muy[i,t], pre_sig2)
          muy[i,t] <- LS[i,1]+(t-1)*LS[i,2]
        }
        LS[i,1:2] ~ dmnorm(muLS[Z[i],1:2], Inv_cov[1:2,1:2])
      }

      #===========================
      # prior specification
      #===========================
      # --------- w ---------
        w[1:K] ~ ddirich(alpha[1:K])
        for(k in 1:K) {alpha[k] <- 1}

      # --------- muLS ---------
      for(k in 1:K) {
        # normal prior for mean of latent intercept
        muLS[k,1] ~ dnorm(0,0.001)
        # normal prior for mean of latent slope
        muLS[k,2] ~ dnorm(0,0.001)
      }

      # --------- Inv_cov ---------
      # Wishart prior for precision matrix
      Inv_cov[1:2,1:2] ~ dwish(R[1:2,1:2],2)
      Cov_b[1:2,1:2] <- inverse(Inv_cov[1:2,1:2])
      R[1,1] <- 1
      R[2,2] <- 1
      R[2,1] <- R[1,2]
      R[1,2] <- 0

      # --------- pre_sig2 ---------
      pre_sig2 ~ dgamma(0.1,0.1)   # gamma prior for precision
      sig2 <- 1/pre_sig2
}"
```

Step 2: Specify initial values

```
# Automatically generated initial values
inits <- list(".RNG.name"="base::Wichmann-Hill", ".RNG.seed"=111)
```

Step 3: Fit the model via JAGS

```
# Bundle data for JAGS
Dat <- list("y"=dat.gmm$y, "Time"=4, "N"=200, "K"=3)
# Run the analysis
set.seed(1234)
out_gmm <- run.jags(model=gmm, monitor=c('w','muLS','Cov_b','sig2'),
          data=Dat, inits=inits, method="rjags", n.chains=1,
          adapt=1000, burnin=1000, sample=10000, thin=1)
```

Step 4: Inspect label switching Similarly, no signs of label switching are found in Figure 9.

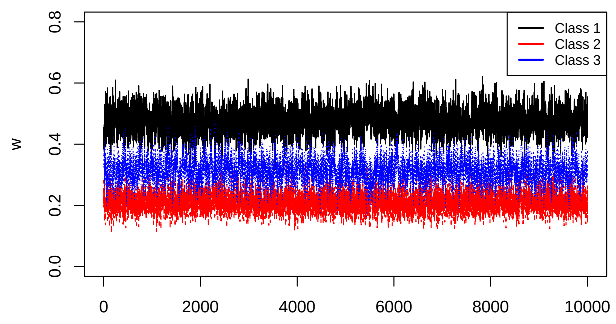


Figure 9. Trace plot of class weights for inspecting label switching.

Step 5: Adaption and burn-in

```
na <- nb <- 1000
out.burn <- window(out_gmm$mcmc, start=na+nb+1)
```

Step 6: Convergence diagnostic via Geweke's test

```
con.diag <- geweke.diag(out.burn)
flag.noncov <- which(abs(con.diag[[1]]$z)>2)
```

Step 7: Summarize posterior MCMC samples

```
sum.stats <- summary(out.burn)         # summary statistics
post.means <- sum.stats$statistics[,1] # posterior means
sum.stats[[1]]                         # display part 1 of the
                                       # summary statistics
```

```
##                   Mean         SD     Naive SE Time-series SE
## w[1]        0.47670814 0.04073104 0.0004073104   0.0012239491
## w[2]        0.21085352 0.03102677 0.0003102677   0.0004599641
## w[3]        0.31243835 0.03959052 0.0003959052   0.0011664481
## muLS[1,1]   1.93669961 0.08287253 0.0008287253   0.0028227498
## muLS[2,1]   5.97551073 0.12216168 0.0012216168   0.0028493240
## muLS[3,1]   3.94876895 0.12075438 0.0012075438   0.0043165390
## muLS[1,2]   0.03924477 0.04421641 0.0004421641   0.0012517566
## muLS[2,2]   0.29034785 0.06514009 0.0006514009   0.0011798124
## muLS[3,2]  -0.38549236 0.05797084 0.0005797084   0.0015226840
## Cov_b[1,1]  0.26303998 0.05820339 0.0005820339   0.0020464038
## Cov_b[2,1]  0.03346480 0.02181222 0.0002181222   0.0007339775
## Cov_b[1,2]  0.03346480 0.02181222 0.0002181222   0.0007339775
## Cov_b[2,2]  0.10267863 0.01584738 0.0001584738   0.0003574567
## sig2        0.26908279 0.01841780 0.0001841780   0.0003431461
```

Step 8: Plot class profiles for interpretation The profiles of the classes are displayed in Figure 10.

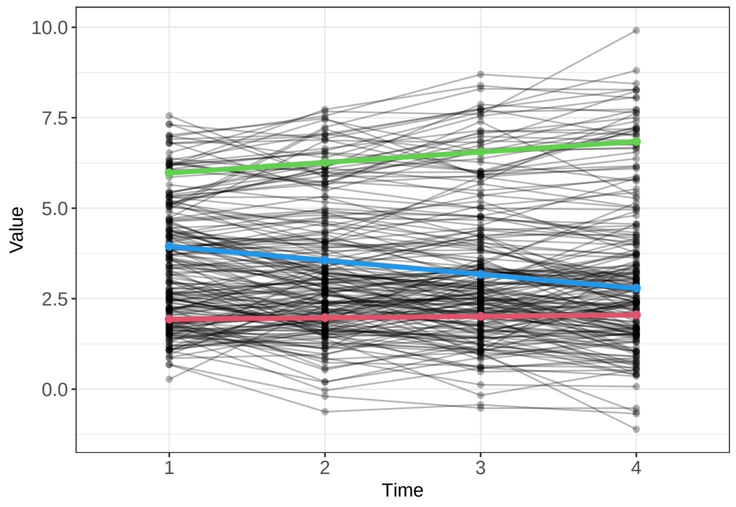


Figure 10. Scatter line plot of growth trajectories. Black thin lines correspond to individiual trajectories, colored thick lines correspond to the estimated growth trajectories representing the three classes.

8 Reporting the Results for LCA

Weller and Faubert (2020) provides guidelines on the information that should be included in an LCA report:

(1) Substantive theories guiding the choice of models to be evaluated are synthesized
(2) Manifest variables are defined and their appropriateness is justified
(3) Report data characteristics (e.g., descriptive statistics, missing data)
(4) Provide statistical software used and the version number
(5) Estimation method is discussed
(6) Criteria used for selecting class model, both statistical (e.g., BIC, aBIC, CAIC) and substantive
(7) Tabulate at least two fit indices, entropy, and smallest average latent class posterior probability

(8) Latent class proportions and conditional probabilities are reported and displayed (e.g., line plot)
(9) Boundary parameter estimates are highlighted and implications are discussed
(10) Meaningfulness of the latent class proportions is considered

Another comprehensive summary table can be found in Hancock et al. (2019, p. 165), which lists key elements that should be addressed in any manuscript's methodological approach to LCA. In addition, Depaoli (2021) provides a template for how to write up Bayesian LCA results for an empirical study (see Section 9.7, p. 340).

9 Discussion

LCA is a popular technique that allows researchers to cluster individuals into latent classes based on response patterns to manifest variables. In this tutorial, LCA is described in a pedagogical manner to address the main challenge confronting applied researchers—how to transition from statistical modeling to computer programming. Although we demonstrated both the conventional LCA and its Bayesian counterpart, we put emphasis on the Bayesian side. The Bayesian framework can be advantageous for estimating LCA, particularly when sample sizes are relatively small. The use of priors can improve the ability to obtain viable and interpretable results. Nonetheless, the Bayesian approach is not exempt from pitfalls. For instance, the label switching phenomenon makes the generated MCMC samples non-identifiable and thus complicates the posterior inference. Additionally, we highlighted other issues that one should be aware of when performing LCA, including boundary estimates and quantitatively different classes.

The basic LCA for binary items along with the real data example should provide the foundation for the readers to move on smoothly to the two more advanced extensions: mixed-mode LCA and latent growth curve mixture model. In the **JAGS** implementation, we decomposed these modeling approaches into eight steps. Such a step-by-step procedure should allow the readers to apply similar models in practice without much difficulty. We hope that this tutorial serves as an approachable entry to a particular context of the extensive Bayesian statistics literature.

Note

Supplementary materials can be downloaded here: `https://doi.org/10.35566/jbds/v2n2/qiu`.

References

Asparouhov, T., & Muthen, B. (2014). Auxiliary variables in mixture modeling: Three-step approaches using mplus. *Structural Equation Modeling*, *21*(3), 329—341. doi: https://doi.org/10.1080/10705511.2014.915181

Cloitre, M., Garvert, D. W., Weiss, B., Carlson, E. B., & Bryant, R. A. (2014). Distinguishing PTSD, complex PTSD, and Borderline Personality Disorder: A latent class analysis. *European Journal of Psychotraumatology*, *5*(1), 25097–25010. doi: https://doi.org/10.3402/ejpt.v5.25097

Collins, L. M., & Lanza, S. T. (2010). *Latent class and latent transition analysis: with applications in the social behavioral, and health sciences.* New Jersey: Wiley.

Depaoli, S. (2021). *Bayesian structural equation modeling.* New York: Guilford.

Hancock, G. R., Harring, J., & Macready, G. B. (2019). *Advances in latent class analysis: A festschrift in honor of C. Mitchell Dayton.* North Carolina: Information Age Publishing.

Hancock, G. R., Stapleton, L. M., & Mueller, R. O. (2019). *The reviewer's guide to quantitative methods in the social sciences* (2nd ed.). New York: Routledge.

Harrington, J. M., Dahly, D. L., Fitzgerald, A. P., Gilthorpe, M. S., & Perry, I. J. (2014). Capturing changes in dietary patterns among older adults: A latent class analysis of an ageing irish cohort. *Public Health Nutrition*, *17*(12), 2674–2686. doi: https://doi.org/10.1017/S1368980014000111

Haughton, D., Legrand, P., & Woolford, S. (2009). Review of three latent class cluster analysis packages: Latent Gold, poLCA, and MCLUST. *The American Statistician*, *63*(1), 81—91. doi: https://doi.org/10.1198/tast.2009.0016

Linzer, D. A., & Lewis, J. (2011). poLCA: An R package for polytomous variable latent class analysis. *Journal of Statistical Software*, *42*(10). doi: https://doi.org/10.18637/jss.v042.i10

McLachlan, G. J., & Peel, D. (2000). *Finite mixture models.* New Jersey: Wiley.

Merz, E. L., & Roesch, S. C. (2011). Latent profile analysis of the five factor model of personality: Modeling trait interactions. *Personality and Individual Differences*, *51*(8), 915–919. doi: https://doi.org/10.1016/j.paid.2011.07.022

Muthen, L. K., & Muthen, B. O. (1998-2017). *Mplus user's guide (8th ed.).* [Muthen and Muthen.].

O'Hagan, A., Murphy, T. B., Scrucca, L., & Gormley, I. C. (2019). Investigation of parameter uncertainty in clustering using a gaussian mixture model via jackknife, bootstrap and weighted likelihood bootstrap. *Computational Statistics*, *34*(4), 1779–1813. doi: https://doi.org/10.1007/s00180-019-00897-9

Papastamoulis, P. (2016). label.switching: An R package for dealing with the label switching problem in MCMC outputs. *Journal of Statistical Software*, *69*(1), 1–24.

Rosenberg, J., Beymer, P., Anderson, D., van Lissa, C. j., & Schmidt, J. (2018). tidyLPA: An R package to easily carry out latent profile analysis (LPA) using open-source or commercial software. *Journal of Open Source Software*, *3*(30), 978. doi: https://doi.org/10.21105/joss.00978

SAS. (2016). *SAS/SHARE 9.4: User's guide.* SAS Institute Inc.

Scrucca, L., Fop, M., Murphy, T. B., & Raftery, A. E. (2016). mclust 5: Clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, *8*(1), 289—317. doi: https://doi.org/10.32614/RJ-2016-021

STATA. (1985-2019). *Stata user's guide: Release 16.* Stata Press.

Vermunt, J. K., & Magidson, J. (2016). *Upgrade manual for Latent GOLD 5.1.* Statistical Innovations Inc.

Weller, B. E., Bowen, N. K., & Faubert, S. J. (2020). Latent class analysis: a guide to best practice. *Journal of Black Psychology*, *46*(4), 287—311. doi: https://doi.org/10.1177/0095798420930932

Xiao, Y., Romanelli, M., & Lindsey, M. A. (2019). A latent class analysis of health lifestyles and suicidal behaviors among US adolescents. *Journal of Affective Disorders*, *255*, 116–126. doi: https://doi.org/10.1016/j.jad.2019.05.031

Yao, W. (2015). Label switching and its solutions for frequentist mixture models. *Annual Review of Statistics and Its Application*, *85*(5), 1000—1012. doi: https://doi.org/10.1080/00949655.2013.859259

Journal of Behavioral Data Science, 2022, *2* (2), 156–173.
DOI: `https://doi.org/10.35566/jbds/v2n2/shao`

A Tutorial on Bayesian Analysis of Count Data Using JAGS

Sijing (SJ) Shao[1]

Department of Psychology, University of Notre Dame, Notre Dame, USA
`sshao2@nd.edu`

Abstract. In behavioral studies, the frequency of a particular behavior or event is often collected and the acquired data are referred to as count data. This tutorial introduces readers to Poisson regression models which is a more appropriate approach for such data. Meanwhile, count data with excessive zeros often occur in behavioral studies and models such as zero-inflated or hurdle models can be employed for handling zero-inflation in the count data. In this tutorial, we aim to cover the necessary fundamentals for these methods and equip readers with application tools of `JAGS`. Examples of the implementation of the models in `JAGS` from within `R` are provided for demonstration purposes.

Keywords: Count data · Zero-inflation · Poisson regression · ZIP model · Hurdle model

1 Introduction

In behavioral studies, information such as the number of times a certain behavior or event occurs is often collected in order to help understand individuals. Such collected nonnegative and discrete data are typically called count data. While normal distribution is the commonly used distribution in most research, specifying a normal distribution for such outcome variables can be inappropriate for at least two reasons: (1) negative and real expected values in a normal distribution is possible while only nonnegative integer values are allowed in such count data; and (2) the distribution of count data is often positively skewed and its variance usually increases along with its mean, while mean and variance are assumed to be unrelated in normal distributions. Poisson regression is more appropriate than general linear regression for such count data, and it models the non-negative integer responses against linear predictors through a link function.

Meanwhile, count data in behavioral research are often heavily skewed due to large amount of zero responses. The zero responses consist of responses from either the individuals who never engaged in such behaviors or those who have engaged but not currently (Grimm & Stegmann, 2019). For example, in alcohol use disorder (AUD) studies, the number of alcohol drinks is often collected from the participants. A zero response could either from participants who never engaged in drinking behavior, or those who drink but not when they are being sampled. The zero-inflated Poisson (ZIP; Lambert (1992)) model was proposed when zeros are assumed to be from both scenarios while the hurdle model (Mullahy, 1986) is appropriate when zeros are assumed to be from only one source. In this tutorial, Poisson regression models for count data, as well as ZIP and hurdle models for zero-inflated response variables are discussed under the Bayesian framework. Examples of estimating these models with `JAGS` (Plummer, 2003) and `R` (Team, 2013) package `runjags` (Denwood, 2016) are illustrated.

1.1 Poisson Regression

The responses $\mathbf{Y} = (Y_1, .., Y_n)^T$ are count of independent events occur in a fixed time interval for n participants. The likelihood function for each response is specified as:

$$Y_i \sim Poisson(\lambda_i), \lambda_i > 0,$$

where the rate parameter λ_i denotes the average number of count per time interval for each person. The density function can be written as $p(Y_i = k) = \frac{e^{-\lambda_i}\lambda_i^k}{k!}$ for $k > 0$. The parameter λ_i can be modeled as a linear function of a set of predictors X with a log link function such that: $log(\lambda_i) = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + ... + \beta_P X_{Pi}$. The parameter β_0 is the intercept, which is the predicted mean of $exp(Y)$ when X is 0. The parameter β_j is the coefficient corresponding to the changes in predictor X_p: one unit increase in X is associated with the expected change in the outcome $exp(Y)$.

1.2 Zero-inflated Poisson (ZIP) Model

In the ZIP framework, the excess zero observations are from either individuals who never engaged in the behaviors of interest, with probability p_i, or individuals who are part of the Poisson distribution in which zeros are generated from participants who have engaged in the behavior but not when the survey was conducted, with probability $1 - p_i$:

$$Y_i \sim \begin{cases} 0, & \text{with probability } p_i \\ Poisson(\lambda_i), & \text{with probability } 1 - p_i, \end{cases}$$

where i indicates the ith participant. The parameter λ_i is the mean parameter for Poisson distribution and represents the expected event frequency for individual i. Thus, $p(Y_i = 0) = p_i + (1 - p_i) \times \frac{e^{-\lambda_i}\lambda_i^0}{0!} = p_i + (1 - p_i) \times e^{-\lambda_i}$ and $p(Y_i = k) = (1 - p_i)\frac{e^{-\lambda_i}\lambda_i^k}{k!}$ for $k > 0$. Let X be the covariates that affect the Poisson mean and B be the covariates affect the probability p_i through log and $logit$ link functions respectively:

$$\begin{aligned} log(\lambda_i) &= \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + ... + \beta_P X_{Pi} \\ logit(p_i) &= \gamma_0 + \gamma_1 B_{1i} + \gamma_2 B_{2i} + ... + \gamma_J B_{Ji}, \end{aligned}$$

where βs and γs are coefficients for design matrices, which include a column of 1 as intercept and predictors X or B respectively. The interpretation for β_0 and β_1 is similar as in Poisson regression. γ_0 and γ_1 are specific to ZIP model. When B is zero, the average odds for a participant to be in the "zero only" group vs. "Poisson" group is $exp(\gamma_0)$. With one unit increases in B, the odds that a participant would be in the "zero only" group vs. "Poisson" group increases by a factor of $exp(\gamma_1)$.

1.3 Hurdle Model

While there are two types of individuals in ZIP, the hurdle model treats all participants in the same way so that everyone could be engaged in the behavior when the survey was undertaken: they could decide to be engaged in the behavior, and then the intensity of the behavior. Thus, two processes are involved in the hurdle model. For n independent observations Y_i:

$$Y_i \sim \begin{cases} 0, & \text{with probability } p_i \\ \text{truncated Poisson}(\lambda_i) & \text{with probability } 1 - p_i. \end{cases}$$

In contrast to ZIP which includes logistic regression to predict "excess zeros" over and above the zeros predicted by Poisson, hurdle models uses logistic regression to predict zero vs non-zeros. The "hurdle" is used to measure whether a response falls below or above the hurdle (e.g., the hurdle is zero in this case). The positive responses above the hurdle zero are then modeled by other truncated count regressions. In this framework, $p(Y_i = 0) = p_i$ and $p(Y_i = k) = \frac{(1-p_i)(\lambda_i^k e^{-\lambda_i})}{(1-e^{-\lambda_i})k!}$ for $k > 0$. Similar to the ZIP framework, design matrices, which include a column of 1 as intercept and predictors X or B, are associated with $log(\lambda_i)$ and $logit(p_i)$ through coefficients βs and γs respectively. However, the interpretation for γs is slightly different from in ZIP: When B is zero, the average odds for a participant not engaging vs. engaging in the behavior is $exp(\gamma_0)$. With one unit increase in B, the odds that a participant would not be engaging vs. engaging in the behavior increases by a factor of $exp(\gamma_1)$.

2 Model Estimation

2.1 Data Description

The data on number of recreational boating trips to Lake Somerville was collected in 1980. The dataset includes 659 responses from registered leisure boat owners in 23 counties in Texas. Figure 1 reveals its variability from 0 up to around 80 with large amount of zero responses. It clearly suggests that this count variable is not normally distributed. The number of recreational trips is often associated with the annual household income. In this illustrative example, we examine whether income is a predictor of number of recreational boating trips a person took. The income variable measures the annual household income of the respondent (in 1,000 USD) and is centered at its mean for the purpose of interpretability. The data is available in `R` package `AER` (Kleiber & Zeileis, 2008).

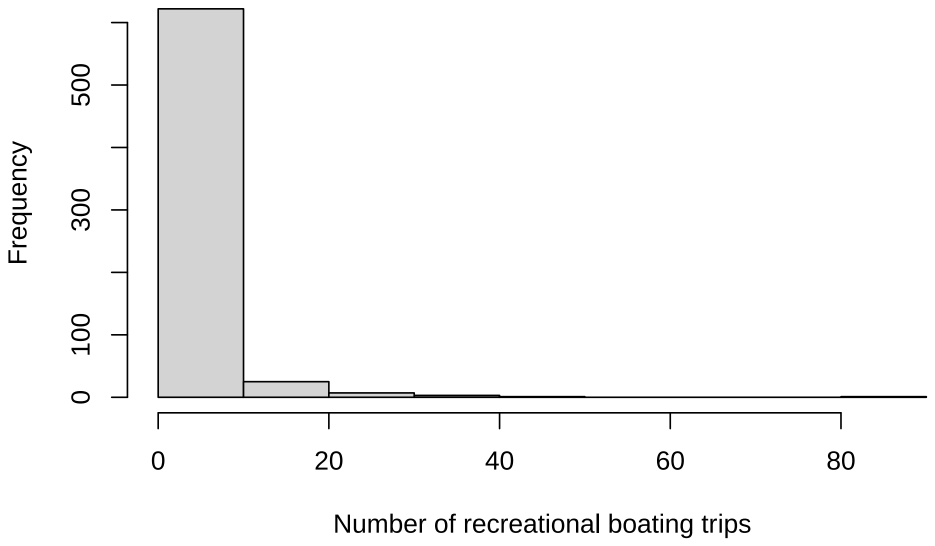


Figure 1. Histogram of the outcome variable.

2.2 Estimation of Poisson Regression in `runjags`

We consider a model in which $log(\lambda)$ is a linear function of income, where λ_i denotes the average number of vacations person i took. The regression equation is:

$$log(\lambda_i) = \beta_0 + \beta_1 \times \text{incomeC}.$$

The model can be estimated in R with package `runjags` and the function `run.jags` is used. It requires a valid model definition, string of monitored variables (`beta0` for β_0 and `beta1` for β_1 in this example), and data, as discussed below.

```
Pois_Est <- run.jags(model = Poisson_Model, monitor =
  c("beta0",  "beta1", "exp_beta0", "exp_beta1"),
  data = data, n.chains = 3,  inits = inits,
  method = "simple", adapt = 1000, burnin = 3000,
  sample = 10000)
```

Some of the important arguments used in the function `run.jags` are:

- model. The model can be specified as a character string, including the likelihood function and initial values. In order to estimate the Poisson regression model using `JAGS`, we first need to specify its likelihood function for all participants and define λ_i with the log link function (see Lines 2 - 6 below). Note the operator " $\sim$" is used to define random variables and it represents "is distributed as". Line 6 `Y[i] ~ dpois(lambda[i])` means that the response `Y[i]` is distributed as a Poisson distribution with rate parameter `lambda[i]`. The operator <- is for the linear function: `log(lambda_i) <- beta0 + beta1*X[i]`. Non-informative priors for β_0 and β_1 can be set as β_0, $\beta_1 \sim N(0, 1000)$. In `JAGS`, a normal distribution is specified as `dnorm(mu, tau)`, with mean `mu` and precision `tau`, where precision is the reciprocal of the variance. Thus, $N(0, 1000)$ is specified as `dnorm(0, 1/1000)` in `JAGS`, see Lines 9 - 10.

```
1  Poisson_Model <- "model{
2     ## Likelihood ##
3     for (i in 1:N){
4      Y[i] ~ dpois(lambda[i])
5      log(lambda[i]) <- beta0 + beta1*X[i]
6     }
7
8     ## priors for coefficients
9      beta0 ~ dnorm(0, 1/1000)
10     beta1 ~ dnorm(0, 1/1000)
11
12    ## exponentiate the paramters
13     exp_beta0 <- exp(beta0)
14     exp_beta1 <- exp(beta1)
15 }"
```

- monitor. The parameters to be estimated are defined in a character string. Since the coefficient parameters are in log odds scale in Poisson regressions, their exponentiated parameters should be obtained for interpretation. In`JAGS`, the exponentiated parameters `exp_beta0` for $exp(\beta_0)$ and `exp_beta1` for $exp(\beta_1)$ can be sampled directly. `exp_beta0` and `exp_beta1` need to specified in `monitor` argument as well as in `model`.
- inits. We need to prepare initial values for `beta0` and `beta1` as shown below. A set of initial values is specified as a list regarding the parameters to be estimated. When multiple chains are generated for convergence diagnosis, a nested list using the `inits` argument with length equal to the number of chains should be specified. In this example, three sets of initial values are specified since three chains are used for convergence diagnosis.

```
inits <- list(list(beta0 = rnorm(1, 0, 0.1),
                   beta1 = rnorm(1, 0, 0.1)),
              list(beta0 = 1, beta1 = 1),
              list(beta0 = -1, beta1 = -1))
```

– data. The variables from data are specified as a list and passed into the data used in `JAGS`, with the argument `data`. The outcome variable Y in `Poisson_Model` is the "trips" variable from the raw data `dat`, denoted as `Y = dat$trips`; the predictor X is the centered "income" variable, denoted as `X = dat$incomeC`. In `Poisson_Model`, `N` is the total sample size and need to be defined as `N = nrow(dat)`.

```
data <- list(Y = dat$trips, X = dat$incomeC,
             N = nrow(dat))
```

– n.chains. Multiple chains can be generated for convergence diagnostic. In this example, three chains were simulated and denoted as `n.chain = 3`. More chains will cause the simulation to run more slowly.
– method. A number of simulation methods are provided in `JAGS`. `simple` is specified here since the model in the illustration example is relatively simple. When more simulation time is possible, other methods allowing parallelisation should be considered.
– adapt. A adaption process is often needed for MCMC samplers in `JAGS` to sample the posteriors more efficiently. The default is 1000 iterations.
– burnin. MCMC samplers often take a finite number of iterations to find the region of posterior probability and this portion of chains should be discarded for inference. The default is 4000 iterations. `burnin` and `adapt` should be specified separately.
– sample. The total number of MCMC samples for each chain can be specified. The default is 10,000 iterations.

Convergence Diagnosis Three Markov chains are obtained with three different set of initial values. The traceplots of the Markov chains for β_0 and β_1 and their exponential values are in Figure 2. Since there is no clear trend in either of the plots and three chains are mixed well, it suggests that convergence is achieved. In addition, the potential scale reduction factor (psrf; (Gelman & Rubin, 1992)) in Table 1 are close to 1, suggesting again that convergence has been reached.

Interpretation The exponentiated coefficients along with the HPD intervals for β_0 and β_1 are shown in Table 1. When the household income is at the average level (3,853 USD), the expected number of boat trips took by the respondents is approximately 2 (2.21) times.

Table 1. Bayesian parameter estimates from Poisson regression

	Mean	SD	Lower 95	Upper 95	psrf
beta0	0.79	0.03	0.74	0.84	1.0002
beta1	-0.10	0.02	-0.13	-0.07	1.0004
exp_beta0	2.21	0.06	2.09	2.32	1.0002
exp_beta1	0.90	0.01	0.88	0.93	1.0004

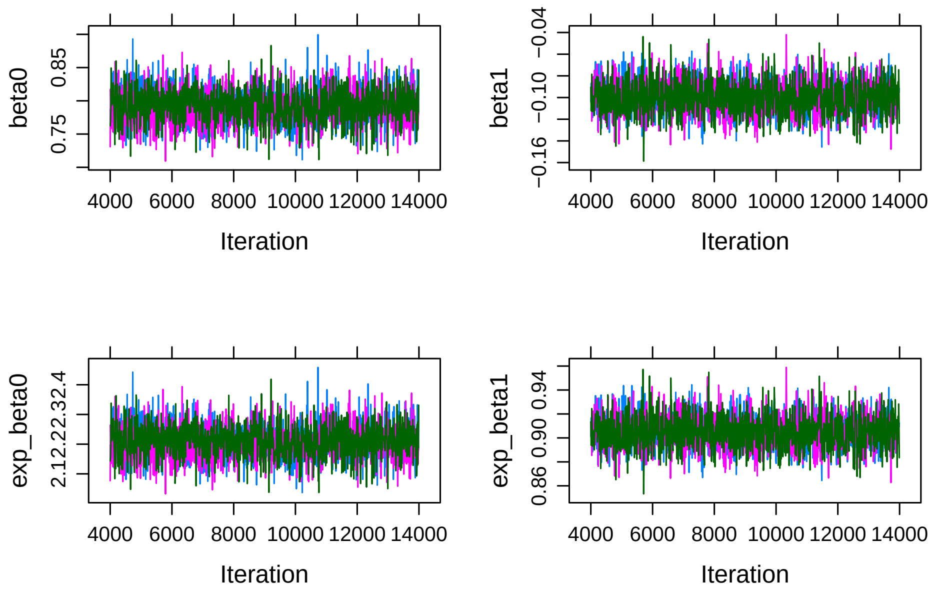


Figure 2. Traceplots of beta0 and beta1 from Poisson regression.

In Table 1, “lower 95” is the 2.5 percentile of the HPD interval and “upper 95” is the 95 percentile. Since 1 is not included in the HPD interval for $exp(\beta_1)$ [0.88, 0.93], the predictor income is statistically associated with number of boat trips took by the respondents. With every $1000 increases in the annual household income, the expected number of boat trips taken by the respondents decreases by 10% (1- (exp(-0.1)) ×100%) on average.

2.3 Estimation of ZIP in `runjags`

The arguments `inits`, `data`, and `model` used in function `run.jags` for the ZIP model are specified in similar ways as in Poisson regression.

```
ZIP_Est <- run.jags(model = ZIP_model,
  monitor = c("beta0", "beta1", "gamma0", "gamma1"),
  data = data, n.chains = 3, inits = inits,
  method = "simple", adapt = 1000, burnin = 3000,
  sample = 10000, keep.jags.files = T, tempdir = T)
```

Similar to estimating Poisson regression in `JAGS`, both likelihood function and prior for parameters are specified first in `ZIP_model`.The likelihood function is defined in Lines 2 - 9. In ZIP, the probability of a zero response coming from the excessive zeros, which are from the group of respondents who never took a boat trip (`W[i] = 0`), is p_i. The probability of a zero response generated from sampling zeros, who usually take boat trips but not when they are being sampled is $1 - p_i$. The sampling zeros are zeros that are generated from the Poisson distribution, denoted as `W[i] = 1`. `W` is a latent Bernoulli random variable and is related to predictor centered income with logit link function:

$$logit(p_i) = \gamma_0 + \gamma_1 \times \text{incomeC}.$$

In another word, when `W[i]` is 0, `W[i]*mu[i]` or `lambda[i]` becomes 0. `Y[i]` is generated from the excessive zeros, which is the group of respondents who never took a boat trip. When `W[i]` is 1, `Y[i]` is generated from the Poisson distribution with rate parameter `mu[i]`. The regression equation for λ_i is as same as in Poisson regression:

$$log(\lambda_i) = \beta_0 + \beta_1 \times \text{incomeC}.$$

Note that the covariates for $logit(p_i)$ and $log(\lambda_i)$ can be the same. For simplicity, we use centered income as the predictor for both. Non-informative prior $N(0, 10000)$ is specified for the four estimated parameters β_0, β_1, γ_0, and γ_1 in Lines 12 - 15.

```
1 ZIP_model <- "model{
2  ## likelihood
3  for (i in 1:N){
4   Y[i] ~ dpois(lambda[i])
5   lambda[i] <- W[i]*mu[i]
```

```
  W[i] ~ dbern(1-p[i])
  log(mu[i]) <- beta0 + beta1*X[i]
  logit(p[i]) <- gamma0 + gamma1*B[i]
 }

 ## prior
  beta0 ~ dnorm(0, 1/10000)
  beta1 ~ dnorm(0, 1/10000)
  gamma0 ~ dnorm(0, 1/10000)
  gamma1 ~ dnorm(0, 1/10000)

 ## exponentiate the paramters
  exp_beta0 <- exp(beta0)
  exp_beta1 <- exp(beta1)
  exp_gamma0 <- exp(gamma0)
  exp_gamma1 <- exp(gamma1)
}"
```

Initial values for `gamma0` and `gamma1` are set in the same way as `beta0` and `beta1`. One new variable `W` is introduced and binary initial values of it are generated. The length of `W` is the total sample size for the data. See Lines 1 - 11 for details. The data specification for ZIP is as same as for Poisson regression as the same dataset is used.

```
1  W <- dat$trips
2  W[dat$trips>0] <- 1
3
4  inits <- list(list(beta0 = rnorm(1, 0, 0.1),
5                     beta1 = rnorm(1, 0, 0.1),
6                     gamma0 = rnorm(1, 0, 0.1),
7                     gamma1 = rnorm(1, 0, 0.1), W = W),
8                list(beta0 = 1, beta1 = 1, gamma0 = 1,
9                     gamma1 = 1, W = W),
10               list(beta0 = -1, beta1 = -1, gamma0 = 1,
11                    gamma1 = 1, W = W))
12
13 data <- list(Y = dat$trips, X = dat$incomeC,
14   B = dat$incomeC, N = nrow(dat))
```

Convergence Diagnosis The methods for convergence diagnosis for ZIP model is as same as for Poisson regression models. Thus, the details are omitted here.

Interpretation The exponentiated coefficients and their HPD intervals for the four estimated parameters are shown in Table 2. The results suggest that for a respondent from a household with average income, the odds of a zero response collected from this person indicating that s/he never went on a boat trip vs. s/she have taken a boat trip but not when being sampled is 1.71. This effect is significant since its HPD interval [1.45, 2] does not contains 1. In addition, when the annual household income increases by 1000 USD, the odds of a zero response being generated from these who never went on a boat trip vs. those who usually took a boat trip but not when being sampled decreases by 3% ($(1 - 0.97) \times 100\%$). This effect is not significant since its HPD interval [0.88, 1.05] contains 1.

Meanwhile, for these who have taken a boat trip but not when being sampled, an increase of $1000 in annual household income is associated with 13% ($(1 - 0.87) \times 100\%$) less average number of boat trips taken by the respondents. This effect is significant since the corresponding HPD interval [0.84, 0.9] does not contain 1.

Table 2. Bayesian parameter estimates from the ZIP model

	Mean	SD	Lower 95	Upper 95	psrf
beta0	1.78	0.03	1.73	1.84	1.0002
beta1	-0.14	0.02	-0.18	-0.11	1.0001
gamma0	0.53	0.08	0.38	0.69	1.0001
gamma1	-0.03	0.05	-0.12	0.05	1.0000
exp_beta0	5.95	0.16	5.63	6.26	1.0002
exp_beta1	0.87	0.02	0.84	0.90	1.0001
exp_gamma0	1.71	0.14	1.45	2.00	1.0001
exp_gamma1	0.97	0.04	0.88	1.05	1.0000

2.4 Estimation of Hurdle Models in `runjags`

In contrast to ZIP where both count and binary parts generate zeros, only the binary part modeled by logistic function in hurdle models generates zeros. The nonzero responses are assumed to be from a truncated Poisson distribution. Zero trick is used when setting up the Bayesian model in `runjags`. The details of zero trick approach are discussed in (Ntzoufras, 2011) and the code for the likelihood function specification is in Lines 2 - 14. `C <- 10000` is specified for the zero trick to make `-ll[i] + C` greater than 0. A dummy variable `z[i]` is created so that it is 0 when `Y[i]` is smaller than 0.0001 and is 1 otherwise.

The log likelihood of the truncated Poisson distribution `truncPois[i]` is defined in Lines 6 - 7. The total likelihood function is the sum of `z[i]*(log(1-p[i]) + truncPois)` and `(1-z[i])*log(p[i])`. When `z[i]` is 0 (`Y[i]` is 0, or `Y[i]` is from the zero-only group), the total likelihood function is `log(p[i])`; when `z[i]` is 1 (`Y[i]` is positive, or `Y[i]` is from the truncated Poisson group), the total likelihood function is `log(1-p[i]) + truncPois`.

Non-informative priors $N(0, 10000)$ is specified in Lines 18 - 21 for the four parameters β_0, β_1, γ_0, and γ_1.

```
hurdle_model <- "model{
 # likelihood
 C <- 10000
 for (i in 1:N){
  zeros[i] ~ dpois(-ll[i] + C)
  truncPois[i] <- Y[i]*log(mu[i]) - mu[i]
     - (log(1-exp(-mu[i])) + logfact(Y[i]))

  l1[i] <- (1-z[i])*log(p[i])
  l2[i] <- z[i]*(log(1-p[i]) + truncPois[i])
  ll[i] <- l1[i] + l2[i]

  log(mu[i]) <- beta0 + beta1*X[i]
  logit(p[i]) <- gamma0 + gamma1*B[i]
 }

 # prior
  beta0 ~ dnorm(0, 1/10000)
  beta1 ~ dnorm(0, 1/10000)
  gamma0 ~ dnorm(0, 1/10000)
  gamma1 ~ dnorm(0, 1/10000)
}"
```

Similar to ZIP, the initial values for the four parameters β_0, β_1, γ_0, and γ_1 are set as below. A column of zeros is added to data for the zero trick approach. Values of `z[i]` are generated from raw data and are provided to the argument `data`.

```
inits <- list(list(beta0 = rnorm(1, 0, 0.1),
                   beta1 = rnorm(1, 0, 0.1),
                   gamma0 = rnorm(1, 0, 0.1),
                   gamma1 = rnorm(1, 0, 0.1)),
              list(beta0 = 1, beta1 = 1, gamma0 = 1,
                   gamma1 = 1),
              list(beta0 = -1, beta1 = -1, gamma0 = 1,
                   gamma1 = 1))
z<-dat$trips
z[dat$trips > 0] <- 1
data <- list(Y = dat$trips, X = dat$incomeC,
             B = dat$incomeC, N = nrow(dat),
             z = z, zeros = rep(0, nrow(dat)))
```

Convergence Diagnosis The methods for convergence diagnosis for hurdle model are the same for Poisson regression models. Thus, the details are omitted here.

Interpretation The exponentiated coefficients and the corresponding HPD intervals are presented in Table 3. For a respondent from a household with average income, the odds of this person not have been to vs. have been to a boat trip is 1.73. This is significant since its HPD interval [1.47, 2.01] does not contain 1. The odds decrease by 2% ($(1 - 0.98) \times 100\%$) when the average house hold income increases by \$1000. This effect is not significant since its HPD interval [0.9, 1.07] contains 1.

Meanwhile, for those who have taken a trip when being sampled, an increase of \$1000 in annual household income is associated with 13% ($(1 - 0.87) \times 100\%$) less average number of boat trips taken by the respondents. This effect is significant since the corresponding HPD interval [0.84, 0.9] does not contain 1.

Table 3. Bayesian parameter estimates from the hurdle model

	Mean	SD	Lower 95	Upper 95	psrf
beta0	1.78	0.03	1.73	1.84	1.0002
beta1	-0.14	0.02	-0.18	-0.11	1.0003
gamma0	0.55	0.08	0.39	0.71	1.0003
gamma1	-0.02	0.04	-0.10	0.07	1.0002
exp_beta0	5.96	0.16	5.65	6.27	1.0002
exp_beta1	0.87	0.02	0.84	0.90	1.0003
exp_gamma0	1.73	0.14	1.47	2.01	1.0003
exp_gamma1	0.98	0.04	0.90	1.07	1.0001

3 Discussion

This tutorial covered methods handling count data and how these methods can be estimated in Bayesian framework with `runjags`. When the data is positively skewed with zero inflation, ZIP and hurdle models can be considered to handle such scenarios. Even though ZIP and hurdle models have been employed interchangeably in psychological research, they are described in two distinct frameworks: ZIP is a mixture model in which zeros can be generated from both Poisson and Bernoulli distributions while hurdle is a two-part model separating zeros from positive responses. While the output tables for ZIP and hurdle models suggest that the results are very similar, the interpretation of the models differs. Researchers should be careful with the choice of methods when working with zero-inflated data.

Furthermore, both ZIP and hurdle models provide more information than Poisson model when zero-inflation is present in the data. For example, income is associated with the odds of zero responses being collected from responders never went vs. have been on a boat trip but not when being sampled in ZIP. At the same time, income is associated with the odds of a person have not been to vs. have been to a boat trip. However, this information is not provided in Poisson models. In addition, even though the estimated βs are similar in all three models, the estimated β in Poisson model is larger than the values estimated in the ZIP and hurdle models.

This tutorial serves for the purpose of illustrating how the models can be estimated with `runjags`. Important topics such as model selections or other distributions handling count data are not covered in this paper. Readers can refer to Feng (2021) for a comprehensive comparison between ZIP and hurdle models handling zero-inflated count data.

References

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*, 1–25. doi: https://doi.org/10.18637/jss.v071.i09

Feng, C. X. (2021). A comparison of zero-inflated and hurdle models for modeling zero-inflated count data. *Journal of statistical distributions and applications*, *8*(1), 1–19. doi: https://doi.org/10.1186/s40488-021-00121-4

Gelman, A., & Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical science*, 457–472. doi: https://doi.org/10.1214/ss/1177011136

Grimm, K. J., & Stegmann, G. (2019). Modeling change trajectories with count and zero-inflated outcomes: Challenges and recommendations. *Addictive Behaviors*, *94*, 4–15. Retrieved 2022-09-04, from `https://linkinghub.elsevier.com/retrieve/pii/S0306460318310177` doi: https://doi.org/10.1016/j.addbeh.2018.09.016

Kleiber, C., & Zeileis, A. (2008). *Applied econometrics with R*. Springer-Verlag. Retrieved from `https://CRAN.R-project.org/package=AER`

Lambert, D. (1992). Zero-inflated Poisson regression, with an application to defects in manufacturing. *Technometrics : a journal of statistics for the physical, chemical, and engineering sciences*, *34*(1), 1–14. doi: https://doi.org/10.2307/1269547

Mullahy, J. (1986). Specification and testing of some modified count data models. *Journal of econometrics*, *33*(3), 341–365. doi: https://doi.org/10.1016/0304-4076(86)90002-3

Ntzoufras, I. (2011). *Bayesian modeling using WinBUGS*. John Wiley & Sons.

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In *Proceedings of the 3rd international workshop on distributed statistical computing* (Vol. 124, pp. 1–10).

Team, R. C. (2013). *R: A language and environment for statistical computing.* Vienna, Austria.

Appendix A Supplemental Material

```
########### Get the data ready for analysis ###########

library(AER)
data("RecreationDemand")
dat<-RecreationDemand
dat$incomeC <- dat$income - mean(dat$income)
hist(dat$trips, main = "",
     xlab = "Number of recreational boating trips")

############### load packages ###############
library(runjags)
library(kableExtra)


############### analyses ###############

###### Poisson ######
Poisson_Model <- "model{
   ## Likelihood ##
   for (i in 1:N){
    Y[i] ~ dpois(lambda[i])
    log(lambda[i]) <- beta0 + beta1*X[i]
   }

   ## priors for coefficients
    beta0 ~ dnorm(0, 1/10000)
    beta1 ~ dnorm(0, 1/10000)

   ## exponentiate the paramters
   exp_beta0 <- exp(beta0)
   exp_beta1 <- exp(beta1)
}"

inits <- list(list(beta0 = rnorm(1, 0, 0.1),
              beta1 = rnorm(1, 0, 0.1)),
              list(beta0 = 1, beta1 = 1),
              list(beta0 = -1, beta1 = -1))

data <- list(Y = dat$trips, X = dat$incomeC,
             N = nrow(dat))

Pois_Est <- run.jags(model = Poisson_Model,
  monitor = c("beta0", "beta1", "exp_beta0",
  "exp_beta1"), data = data, n.chains = 3,
  inits = inits, method = "simple", adapt = 1000,
  burnin = 3000, sample = 10000)


res11<-cbind(round(Pois_Est$HPD[,c(1,3)],2),
  round(Pois_Est$summary$statistics[,1:2],2),
  round(Pois_Est$psrf$psrf[,1],4))
colnames(res11) <- c("Lower 95", "Upper 95",
  "Mean", "SD", "psrf")
kable(res11, caption = "Poisson runjags
  Output", "simple")
par(mfrow = c(1, 2))
plot(Pois_Est, plot.type = "trace")


###### ZIP ######
ZIP_model <- "model{
 ## likelihood
 for (i in 1:N){
  Y[i] ~ dpois(lambda[i])
  lambda[i] <- W[i]*mu[i]
  W[i] ~ dbern(1-p[i])
  log(mu[i]) <- beta0 + beta1*X[i]
  logit(p[i]) <- gamma0 + gamma1*B[i]
 }

 ## prior
  beta0 ~ dnorm(0, 1/10000)
  beta1 ~ dnorm(0, 1/10000)
  gamma0 ~ dnorm(0, 1/10000)
  gamma1 ~ dnorm(0, 1/10000)

  ## exponentiate the paramters
  exp_beta0 <- exp(beta0)
  exp_beta1 <- exp(beta1)
  exp_gamma0 <- exp(gamma0)
  exp_gamma1 <- exp(gamma1)
}"

W <- dat$trips
W[dat$trips>0] <- 1

inits <- list(list(beta0 = rnorm(1, 0, 0.1),
              beta1 = rnorm(1, 0, 0.1),
               gamma0 = rnorm(1, 0, 0.1),
               gamma1 = rnorm(1, 0, 0.1),
               W = W), list(beta0 = 1, beta1 = 1,
               gamma0 = 1, gamma1 = 1, W = W),
               list(beta0 = -1, beta1 = -1,
               gamma0 = 1,
               gamma1 = 1, W = W))

data <- list(Y = dat$trips, X = dat$incomeC,
             B = dat$incomeC, N = nrow(dat))

ZIP_Est <- run.jags(model = ZIP_model, monitor =
   c("beta0", "beta1", "gamma0", "gamma1",
   "exp_beta0", "exp_beta1", "exp_gamma0",
   "exp_gamma1"), data = data, n.chains = 3,
    inits = inits, method = "simple", adapt = 1000,
    burnin = 3000, sample = 10000,
    keep.jags.files = T, tempdir = T)
res2<-cbind(round(ZIP_Est$HPD[,c(1,3)],2),
  round(ZIP_Est$summary$statistics[,1:2],2),
  round(ZIP_Est$psrf$psrf[,1],4))
colnames(res2) <- c("Lower 95", "Upper 95", "Mean",
  "SD", "psrf")
res22<-round(exp(res2[,c(1, 2, 3)]),2)

kable(res22, caption = "ZIP runjags Exponentiated
  Output", "simple")
par(mfrow = c(1, 2))
plot(ZIP_Est, plot.type = "trace")


###### Hurdle ######
hurdle_model <- "model{
 ## likelihood
 C <- 10000
  for (i in 1:N){
  zeros[i] ~ dpois(-ll[i] + C)
  truncPois[i] <- Y[i]*log(mu[i]) - mu[i] -
        (log(1-exp(-mu[i])) + logfact(Y[i]))

  l1[i] <- (1-z[i])*log(p[i])
  l2[i] <- z[i]*(log(1-p[i]) + truncPois[i])
  ll[i] <- l1[i] + l2[i]

  log(mu[i]) <- beta0 + beta1*X[i]
  logit(p[i]) <- gamma0 + gamma1*B[i]
 }

 ## prior
  beta0 ~ dnorm(0, 1/10000)
  beta1 ~ dnorm(0, 1/10000)
  gamma0 ~ dnorm(0, 1/10000)
  gamma1 ~ dnorm(0, 1/10000)

 ## exponentiate the paramters
  exp_beta0 <- exp(beta0)
  exp_beta1 <- exp(beta1)
  exp_gamma0 <- exp(gamma0)
  exp_gamma1 <- exp(gamma1)
}"

z<-dat$trips
z[dat$trips > 0] <- 1

inits <- list(list(beta0 = rnorm(1, 0, 0.1),
                beta1 = rnorm(1, 0, 0.1),
                gamma0 = rnorm(1, 0, 0.1),
                gamma1 = rnorm(1, 0, 0.1)),
                list(beta0 = 1, beta1 = 1,
                gamma0 = 1, gamma1 = 1),
                list(beta0 = -1, beta1 = -1,
                gamma0 = 1,
                gamma1 = 1))

data <- list(Y = dat$trips, X = dat$incomeC,
                B = dat$incomeC,
                N = nrow(dat), z = z,
                zeros = rep(0, nrow(dat)))

hurdle_Est <- run.jags(model = hurdle_model,
                monitor = c("beta0", "beta1",
                 "gamma0", "gamm a1", "exp_beta0",
                "exp_beta1", "exp_gamma0", "exp_gamma1"),
                data = data, n.chains = 3,
                inits = inits, method = "simple",
                adapt = 1000,  burnin = 3000,
                sample = 10000,
                keep.jags.files = T, tempdir = T)

res33<-cbind(round(hurdle_Est$HPD[,c(1,3)],2),
          round(hurdle_Est$summary$statistics[,1:2],2),
          round(hurdle_Est$psrf$psrf[,1],4))
          colnames(res33) <- c("Mean", "SD",
          "Lower 95", "Upper 95","psrf")
res33<-round(exp(res33[,c(1, 2, 3)]),2)
kable(res33, caption = "Hurdle runjags
  Exponentiated Output", "simple")
plot(hurdle_Est, plot.type = "trace")
```