blavaan: Bayesian Structural Equation Models via Parameter Expansion

Edgar C. Merkle
University of Missouri

Yves Rosseel
Ghent University

Abstract

This article describes **blavaan**, an R package for estimating Bayesian structural equation models (SEMs) via **JAGS** and for summarizing the results. It also describes a novel parameter expansion approach for estimating specific types of models with residual covariances, which facilitates estimation of these models in **JAGS**. The methodology and software are intended to provide users with a general means of estimating Bayesian SEMs, both classical and novel, in a straightforward fashion. Users can estimate Bayesian versions of classical SEMs with **lavaan** syntax, they can obtain state-of-the-art Bayesian fit measures associated with the models, and they can export **JAGS** code to modify the SEMs as desired. These features and more are illustrated by example, and the parameter expansion approach is explained in detail.

Keywords: Bayesian SEM, structural equation models, **JAGS**, MCMC, **lavaan**.

1. Introduction

The intent of **blavaan** is to implement Bayesian structural equation models (SEMs) that harness open source MCMC samplers (in **JAGS**; Plummer 2003) while simplifying model specification, summary, and extension. Bayesian SEM has received increasing attention in recent years, with MCMC samplers being developed for specific priors (e.g., Lee 2007; Scheines, Hoijtink, and Boomsma 1999), specific models being estimated in **JAGS** and BUGS (Lunn, Jackson, Best, Thomas, and Spiegelhalter 2012; Lunn, Thomas, Best, and Spiegelhalter 2000), factor analysis samplers being included in R packages **bfa** (Murray 2014) and **MCMCpack** (Martin, Quinn, and Park 2011), and multiple samplers being implemented in **Mplus** (Muthén and Asparouhov 2012). These methods have notable advantages over analogous frequentist methods, including the facts that estimation of complex models is typically easier (e.g., Marsh, Wen, Hau, and Nagengast 2013) and that estimates of parameter uncertainty do not rely on

asymptotic arguments. Further, in addition to allowing for the inclusion of existing knowledge into model estimates, prior distributions can be used to inform underidentified models, to average across multiple models in a single run (e.g., Lu, Chow, and Loken 2016), and to avoid Heywood cases (Martin and McDonald 1975).

The traditional SEM framework (e.g., Bollen 1989) assumes an $n \times p$ data matrix $\boldsymbol{Y} = (\boldsymbol{y}_1 \ \boldsymbol{y}_2 \ \dots \ \boldsymbol{y}_n)^{\top}$ with n independent cases and p continuous manifest variables. Using the **LISREL** "all y" notation (e.g., Jöreskog and Sörbom 1997), a structural equation model with m latent variables may be represented by the equations

$$\boldsymbol{y} = \boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\eta} + \boldsymbol{\epsilon} \tag{1}$$

$$\boldsymbol{\eta} = \boldsymbol{\alpha} + \boldsymbol{B}\boldsymbol{\eta} + \boldsymbol{\zeta}, \tag{2}$$

where $\boldsymbol{\eta}$ is an $m \times 1$ vector containing the latent variables; $\boldsymbol{\epsilon}$ is a $p \times 1$ vector of residuals; and $\boldsymbol{\zeta}$ is an $m \times 1$ vector of residuals associated with the latent variables. Each entry in these three vectors is independent of the others. Additionally, $\boldsymbol{\nu}$ and $\boldsymbol{\alpha}$ contain intercept parameters for the manifest and latent variables, respectively; $\boldsymbol{\Lambda}$ is a matrix of factor loadings; and $\boldsymbol{B}$ contains parameters that reflect directed paths between latent variables (with the assumption that $(\boldsymbol{I} - \boldsymbol{B})$ is invertible).

In conventional SEMs, we assume multivariate normality of the $\boldsymbol{\epsilon}$ and $\boldsymbol{\zeta}$ vectors. In particular,

$$\boldsymbol{\epsilon} \sim N_p(\boldsymbol{0}, \boldsymbol{\Theta}) \tag{3}$$

$$\boldsymbol{\zeta} \sim N_m(\boldsymbol{0}, \boldsymbol{\Psi}), \tag{4}$$

with the latter equation implying multivariate normality of the latent variables. Taken together, the above assumptions imply that the marginal distribution of $\boldsymbol{y}$ (integrating out the latent variables) is multivariate normal with parameters

$$\boldsymbol{\mu} = \boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\alpha} \tag{5}$$

$$\boldsymbol{\Sigma} = \boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Psi}(\boldsymbol{I} - \boldsymbol{B}^{\top})^{-1}\boldsymbol{\Lambda}^{\top} + \boldsymbol{\Theta}. \tag{6}$$

If one is restricted to conjugate priors, then the Song and Lee (2012) or Muthén and Asparouhov (2012) procedures are often available for simple MCMC estimation of the above model. However, if one wishes to use general priors or to estimate novel models, then there is the choice of implementing a custom MCMC scheme or using a general MCMC program like BUGS, **JAGS**, or Stan (Carpenter, Gelman, Hoffman, Lee, Goodrich, Betancourt, Brubaker, Guo, Li, and Riddell 2017). These options are often time-consuming and difficult to extend to further models. This is where **blavaan** is intended to be helpful: it allows for simple specification of Bayesian SEMs while also allowing the user to extend the original model. In addition to easing Bayesian SEM specification, the package includes a novel approach to **JAGS** model estimation that allows us to handle models with correlated residuals. This approach, further described below, builds on the work of many previous researchers: the approach of Lee (2007) for estimating models in WinBUGS; the approach of Palomo, Dunson, and Bollen (2007) for handling correlated residuals; and the approaches of Barnard, McCulloch, and Meng (2000) and Muthén and Asparouhov (2012) for specifying prior distributions of covariance matrices.

Package **blavaan** is potentially useful for the analyst that wishes to estimate Bayesian SEMs and for the methodologist that wishes to develop novel extensions of Bayesian SEMs. The package is a series of bridges between several existing R packages, with the bridges being

used to make each step of the modeling easy and flexible. Package **lavaan** (Rosseel 2012) serves as the starting point and ending point in the sequence: the user specifies models via **lavaan** syntax, and objects of class **blavaan** make use of many **lavaan** functions. Consequently, writings on **lavaan** (e.g., Beaujean 2014; Rosseel 2012) give the user a good idea about how to use **blavaan**.

Following model specification, **blavaan** examines the details of the specified model and converts the specification to **JAGS** syntax. Importantly, this conversion often makes use of the parameter expansion ideas presented below, resulting in MCMC chains that quickly converge for many models. We employ package **runjags** (Denwood 2016) for sampling parameters and summarizing the MCMC chains. Once **runjags** is finished, **blavaan** organizes summaries and computes a variety of Bayesian fit measures.

2. Bayesian SEM

As noted above, the approach of Lee (2007; see also Song and Lee 2012) involves the use of conjugate priors on the parameters from Equations 1 to 4. Specific priors include inverse gamma distributions on variance parameters, inverse Wishart distributions on unrestricted covariance matrices (typically covariances of exogenous latent variables), and normal distributions on other parameters.

Importantly, Lee assumes that the manifest variable covariance matrix $\mathbf{\Theta}$ and the endogenous latent variable covariance matrix $\mathbf{\Psi}$ are diagonal. This assumption of diagonal covariance matrices is restrictive in some applications. For example, researchers often wish to fit models with correlated residuals, and it has been argued that such models are both necessary and under-utilized (Cole, Ciesla, and Steiger 2007). Correlated residuals pose a difficult problem for MCMC methods because they often result in covariance matrices with some (but not all) off-diagonal entries equal to zero. In this situation, we cannot assign an inverse Wishart prior to the full covariance matrix because this does not allow us to fix some off-diagonal entries to zero. Further, if we assign a prior to each free entry of the covariance matrix and sample the parameters separately, we can often obtain covariance matrices that are not positive definite. This can lead to numerical problems, resulting in an inability to carry out model estimation. Finally, it is possible to specify an equivalent model that possesses the required, diagonal matrices. However, the setting of prior distributions can be unclear here: if the analyst specifies prior distributions for her model of interest, then it may be cumbersome to translate these into prior distributions for the equivalent model.

To address the issue of non-diagonal covariance matrices, Muthén and Asparouhov (2012) implemented a random walk method that is based on work by Chib and Greenberg (1998). This method samples free parameters of the covariance matrix via Metropolis-Hastings steps. While the implementation is fast and efficient, it does not allow for some types of equality constraints because parameters are updated in blocks (either all parameters in a block must be constrained to equality, or no parameters in a block can be constrained). Further, the method is unreliable for models involving many latent variables. Consequently, Muthén and Asparouhov (2012) implemented three other MCMC methods that are suited to different types of models.

In our initial work on package **blavaan**, we developed methodology for fitting models with free residual covariance parameters. We sought methodology that (i) would work in **JAGS**,

(ii) was reasonably fast and efficient (by **JAGS** standards), and (iii) allowed for satisfactory specification of prior distributions. In the next section, we describe the resulting methodology. The methodology can often be used to reliably estimate posterior means to the first decimal place on the order of minutes (often, less than two minutes on desktop computers, but it also depends on model complexity). This will never beat compiled code, but it is typically better than alternative **JAGS** parameterizations that rely on multivariate distributions.

2.1. Parameter expansion

General parameter expansion approaches to Bayesian inference are described by Gelman (2004, 2006), with applications to factor analysis models detailed by Ghosh and Dunson (2009). Our approach here is related to that of Palomo *et al.* (2007), who employ phantom latent variables (they use the term pseudo-latent variable) to simplify the estimation of models with non-diagonal $\boldsymbol{\Theta}$ matrices. This results in a *working model* that is estimated, with the working model parameters being transformed to the *inferential model* parameters from Equations 1 and 2. The use of phantom latent variables in SEM has long been known (e.g., Rindskopf 1984), though the Bayesian approach involves new issues associated with prior distribution specification. This is further described below.

Overview

Assuming v nonzero entries in the lower triangle of $\boldsymbol{\Theta}$ (i.e., v residual covariances), Palomo *et al.* (2007) take the inferential model residual vector $\boldsymbol{\epsilon}$ and reparameterize it as

$$\boldsymbol{\epsilon} = \boldsymbol{\Lambda}_D \boldsymbol{D} + \boldsymbol{\epsilon}^*$$
$$\boldsymbol{D} \sim N_v(\boldsymbol{0}, \boldsymbol{\Psi}_D)$$
$$\boldsymbol{\epsilon}^* \sim N_p(\boldsymbol{0}, \boldsymbol{\Theta}^*),$$

where $\boldsymbol{\Lambda}_D$ is a $p \times v$ matrix with many zero entries, $\boldsymbol{D}$ is a $v \times 1$ vector of phantom latent variables, and $\boldsymbol{\epsilon}^*$ is a $p \times 1$ residual vector. This approach is useful because, by carefully choosing the nonzero entries of $\boldsymbol{\Lambda}_D$, both $\boldsymbol{\Psi}_D$ and $\boldsymbol{\Theta}^*$ are diagonal. This allows us to employ an approach related to Lee (2007), avoiding high-dimensional normal distributions in favor of univariate normals.

Under this working model parameterization, the inferential model covariance matrix $\boldsymbol{\Theta}$ can be re-obtained via

$$\boldsymbol{\Theta} = \boldsymbol{\Lambda}_D \boldsymbol{\Psi}_D \boldsymbol{\Lambda}_D^\top + \boldsymbol{\Theta}^*.$$

We can also handle covariances between latent variables in the same way, as necessary. Assuming m latent variables with w covariances, we have

$$\boldsymbol{\zeta} = \boldsymbol{B}_E \boldsymbol{E} + \boldsymbol{\zeta}^*$$
$$\boldsymbol{E} \sim N_w(\boldsymbol{0}, \boldsymbol{\Psi}_E)$$
$$\boldsymbol{\zeta}^* \sim N_m(\boldsymbol{0}, \boldsymbol{\Psi}^*),$$

where the original covariance matrix $\boldsymbol{\Psi}$ is re-obtained via:

$$\boldsymbol{\Psi} = \boldsymbol{B}_E \boldsymbol{\Psi}_E \boldsymbol{B}_E^\top + \boldsymbol{\Psi}^*.$$

This approach to handling latent variable covariances can lead to slow convergence in models with many (say, > 5) correlated latent variables. In these cases, we have found it better (in **JAGS**) to sample the latent variables from multivariate normal distributions. Package **blavaan** attempts to use multivariate normal distributions in situations where it can, reverting to the above reparameterization only when necessary.

To choose nonzero entries of $\mathbf{\Lambda}_D$ (with the same method applying to nonzero entries of $\boldsymbol{B}_E$), we define two $v \times 1$ vectors $\boldsymbol{r}$ and $\boldsymbol{c}$. These vectors contain the respective row numbers and column numbers of the nonzero, lower-triangular entries of $\mathbf{\Theta}$. For $j = 1, \ldots, v$, the nonzero entries of $\mathbf{\Lambda}_D$ then occur in column j, rows r_j and c_j. Palomo *et al.* (2007) set these nonzero entries equal to 1, which can be problematic if the covariance parameters' posteriors are negative or overlap with zero. This is because the only remaining free parameters are variances, which can only be positive. Instead of 1s, we free the parameters in $\mathbf{\Lambda}_D$ so that they are functions of inferential model parameters. This is further described in the next section. First, however, we give an example of the approach.

Example

Consider the political democracy example of Bollen (1989), which includes eleven observed variables that are hypothesized to arise from three latent variables. The inferential structural equation model affiliated with these data appears as a path diagram (with variance parameters omitted) at the top of Figure 1. This model can be written in matrix form as

$$
\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \end{pmatrix} = \begin{pmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \\ \nu_4 \\ \nu_5 \\ \nu_6 \\ \nu_7 \\ \nu_8 \\ \nu_9 \\ \nu_{10} \\ \nu_{11} \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ \lambda_2 & 0 & 0 \\ \lambda_3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \lambda_5 & 0 \\ 0 & \lambda_6 & 0 \\ 0 & \lambda_7 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & \lambda_5 \\ 0 & 0 & \lambda_6 \\ 0 & 0 & \lambda_7 \end{pmatrix} \begin{pmatrix} \text{ind60} \\ \text{dem60} \\ \text{dem65} \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \\ e_{10} \\ e_{11} \end{pmatrix}
$$

$$
\begin{pmatrix} \text{ind60} \\ \text{dem60} \\ \text{dem65} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ b_1 & 0 & 0 \\ b_2 & b_3 & 0 \end{pmatrix} \begin{pmatrix} \text{ind60} \\ \text{dem60} \\ \text{dem65} \end{pmatrix} + \begin{pmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{pmatrix}.
$$

Additionally, the latent residual covariance matrix $\mathbf{\Psi}$ is diagonal and the observed residual covariance matrix is

$$
\mathbf{\Theta} = \begin{pmatrix}
\theta_{11} & & & & & & & & & & \\
& \theta_{22} & & & & & & & & & \\
& & \theta_{33} & & & & & & & & \\
& & & \theta_{44} & & & & \theta_{48} & & & \\
& & & & \theta_{55} & & \theta_{57} & & \theta_{59} & & \\
& & & & & \theta_{66} & & & & \theta_{6,10} & \\
& & & & \theta_{57} & & \theta_{77} & & & & \theta_{7,11} \\
& & & \theta_{48} & & & & \theta_{88} & & & \\
& & & & \theta_{59} & & & & \theta_{99} & & \theta_{9,11} \\
& & & & & \theta_{6,10} & & & & \theta_{10,10} & \\
& & & & & & \theta_{7,11} & & \theta_{9,11} & & \theta_{11,11}
\end{pmatrix},
$$

where the blank entries all equal zero. The six unique, off-diagonal entries of $\boldsymbol{\Theta}$ make the model difficult to estimate via Bayesian methods.

To ease model estimation, we follow Palomo *et al.* (2007) in reparameterizing the model to the working model displayed at the bottom of Figure 1. The working model is now

$$
\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \end{pmatrix} = \begin{pmatrix} \nu_1 \\ \nu_2 \\ \nu_3 \\ \nu_4 \\ \nu_5 \\ \nu_6 \\ \nu_7 \\ \nu_8 \\ \nu_9 \\ \nu_{10} \\ \nu_{11} \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 \\ \lambda_2 & 0 & 0 \\ \lambda_3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \lambda_5 & 0 \\ 0 & \lambda_6 & 0 \\ 0 & \lambda_7 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & \lambda_5 \\ 0 & 0 & \lambda_6 \\ 0 & 0 & \lambda_7 \end{pmatrix} \begin{pmatrix} \text{ind60} \\ \text{dem60} \\ \text{dem65} \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \lambda_{D1} & 0 & 0 & 0 & 0 & 0 \\ 0 & \lambda_{D2} & \lambda_{D3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \lambda_{D4} & 0 & 0 \\ 0 & \lambda_{D5} & 0 & 0 & \lambda_{D6} & 0 \\ \lambda_{D7} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \lambda_{D8} & 0 & 0 & \lambda_{D9} \\ 0 & 0 & 0 & \lambda_{D10} & 0 & 0 \\ 0 & 0 & 0 & 0 & \lambda_{D11} & \lambda_{D12} \end{pmatrix} \begin{pmatrix} D1 \\ D2 \\ D3 \\ D4 \\ D5 \\ D6 \end{pmatrix} + \begin{pmatrix} e_1^* \\ e_2^* \\ e_3^* \\ e_4^* \\ e_5^* \\ e_6^* \\ e_7^* \\ e_8^* \\ e_9^* \\ e_{10}^* \\ e_{11}^* \end{pmatrix}
$$

$$
\begin{pmatrix} \text{ind60} \\ \text{dem60} \\ \text{dem65} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ b_1 & 0 & 0 \\ b_2 & b_3 & 0 \end{pmatrix} \begin{pmatrix} \text{ind60} \\ \text{dem60} \\ \text{dem65} \end{pmatrix} + \begin{pmatrix} \zeta_1 \\ \zeta_2 \\ \zeta_3 \end{pmatrix}.
$$

The covariance matrix associated with $\boldsymbol{\epsilon}^*$, $\boldsymbol{\Theta}^*$, is now diagonal, which makes the model easier to estimate via MCMC. The latent variable residual, $\boldsymbol{\zeta}$, is maintained as before because its co-variance matrix was already diagonal. In general, we only reparameterize residual covariance matrices that are neither diagonal nor unconstrained.

As mentioned earlier, the difference between our approach and that of Palomo *et al.* (2007) is that we estimate the loadings with D subscripts, whereas Palomo *et al.* (2007) fix these loadings to 1. This allows us to obtain posteriors on residual covariances that overlap with zero or that become negative, as necessary. Estimation of these loadings comes with a cost, however, in that the prior distributions of the inferential model do not immediately translate into prior distributions of the working model. This problem and a solution are further described in the next section.

2.2. Priors on covariances

Due to the reparameterization of covariances, prior distributions on the working model parameters require care in their specification. Every inferential model covariance parameter is potentially the product of three parameters: two representing paths from a phantom latent variable to the variable of interest, and one precision (variance) of the phantom latent variable. These three parameters also impact the inferential model variance parameters. We carefully restrict these parameters to arrive at prior distributions that are both meaningful and flexible.

We place univariate prior distributions on model variance and correlation parameters, with the distributions being based on the inverse Wishart or on other prior distributions described in the literature. This is highly related to the approach of Barnard *et al.* (2000), who separately specify prior distributions on the variance and correlation parameters comprising a covariance matrix (for an overview and related approaches, see Alvarez, Niemi, and Simpson 2014). Our approach is also related to Muthén and Asparouhov (2012), who focus on marginal distributions associated with the inverse Wishart. A notable difference is that we always use
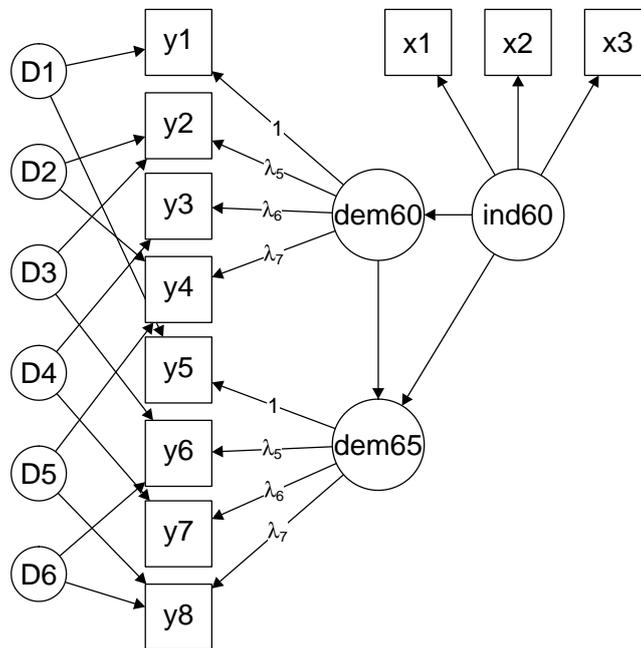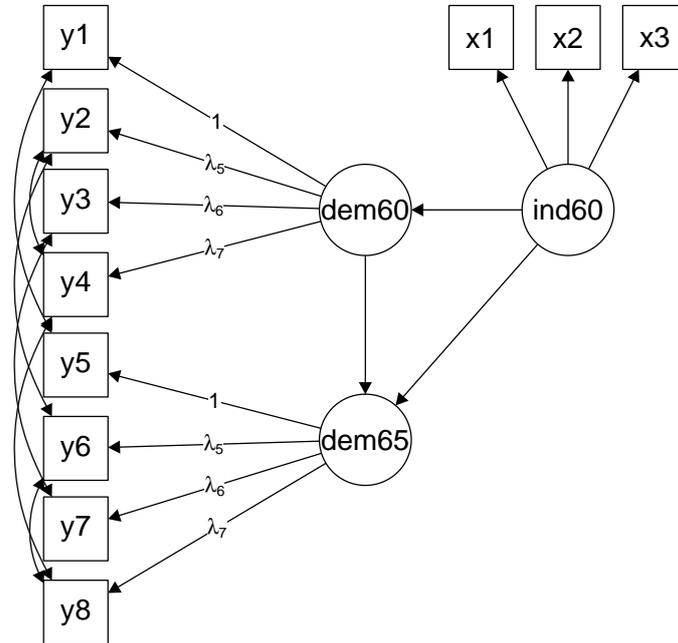
Figure 1: Political democracy path diagrams, original (top) and parameter-expanded version (bottom). Variance parameters are omitted.

proper prior distributions, whereas Muthén and Asparouhov (2012) often use improper prior distributions.

To illustrate the approach, we refer to Figure 2. This figure uses path diagrams to display two observed variables with correlated residuals (left panel) along with their parameter-expanded representation in the working model (right panel). Our goal is to specify a sensible prior distribution for the inferential model's variance/covariance parameters in the left panel, using the parameters in the right panel. The covariance of interest, θ_{12}, has been converted to a normal phantom latent variable with variance ψ_D and two directed paths, λ_1 and λ_2. To implement an approach related to that of Barnard *et al.* (2000) in **JAGS**, we set

$$\psi_D = 1$$
$$\lambda_1 = \sqrt{|\rho_{12}|\theta_{11}}$$
$$\lambda_2 = \text{sign}(\rho_{12})\sqrt{|\rho_{12}|\theta_{22}}$$
$$\theta_{11}^* = \theta_{11} - |\rho_{12}|\theta_{11}$$
$$\theta_{22}^* = \theta_{22} - |\rho_{12}|\theta_{22},$$

where ρ_{12} is the correlation associated with the covariance θ_{12}, and $\text{sign}(a)$ equals 1 if $a > 0$ and -1 otherwise.

Using this combination of parameters, we need only set prior distributions on the inferential model parameters θ_{11}, θ_{22}, and ρ_{12}, with θ_{12} being obtained from these parameters in the usual way. If we want our priors to be analogous to an inverse Wishart prior with d degrees of freedom and diagonal scale matrix $\boldsymbol{S}$, we can set univariate priors of

$$\theta_{11} \sim IG((d-p+1)/2, s_{11}/2)$$
$$\theta_{22} \sim IG((d-p+1)/2, s_{22}/2)$$
$$\rho_{12} \sim \text{Beta}_{(-1,1)}((d-p+1)/2, (d-p+1)/2),$$

where $\text{Beta}_{(-1,1)}$ is a beta distribution with support on $(-1,1)$ instead of the usual $(0,1)$ and p is the dimension of $\boldsymbol{S}$ (typically, the number of observed variables). These priors are related to those used by Muthén and Asparouhov (2012), based on results from Barnard *et al.* (2000). They are also the default priors for variance/correlation parameters in **blavaan**, with $d = (p+1)$ and $\boldsymbol{S} = \boldsymbol{I}$. We refer to this parameterization option as `"srs"`, reflecting the fact that we are dissecting the covariance parameters into standard deviation and correlation parameters.

Barnard *et al.* (2000) avoid the inverse gamma priors on the variance parameters, instead opting for log-normal distributions on the standard deviations that they judged to be more numerically stable. Package **blavaan** allows for custom priors, so that the user can employ the log-normal or others on precisions, variances, or standard deviations (this is illustrated later). This approach is generally advantageous because it allows for flexible specification of prior distributions on covariance matrices, including those with fixed zeros or those where we have different amounts of prior information on certain variance/correlation parameters within the matrix.

While we view the `"srs"` approach as optimal for dealing with covariance parameters here, there exist similar alternative approaches in **JAGS**. For example, we can treat the phantom
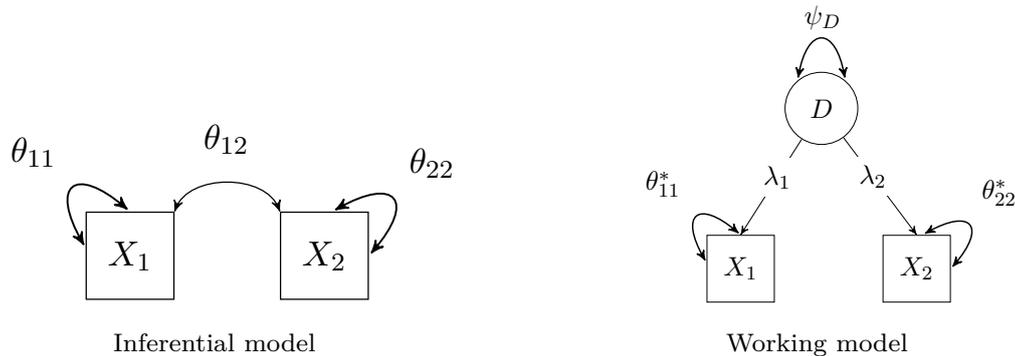
Figure 2: Example of phantom latent variable approach to covariances.

latent variables similarly to the other latent variables in the model, assigning prior distributions in the same manner. This alternative approach, which we call `"fa"` (because the priors resemble traditional factor analysis priors), involves the following default prior distributions on the working model from Figure 2:

$$\psi_D \sim IG(1, .5)$$
$$\lambda_1 \sim N(0, 10^4)$$
$$\lambda_2 \sim N(0, 10^4)$$
$$\theta_{11}^* \sim IG(1, .5)$$
$$\theta_{22}^* \sim IG(1, .5),$$

with the inferential model parameters being obtained in the usual way:

$$\theta_{11} = \theta_{11}^* + \lambda_1^2 \psi_D$$
$$\theta_{22} = \theta_{22}^* + \lambda_2^2 \psi_D$$
$$\theta_{12} = \lambda_1 \lambda_2 \psi_D.$$

The main disadvantage of the `"fa"` approach is that the implied prior distributions on the inferential model parameters are not of a common form. Thus, it is difficult to introduce informative prior distributions for the inferential model variance/covariance parameters. For example, the prior on the inferential covariance (θ_{12}) is the product of two normal prior distributions and an inverse gamma, which can be surprisingly informative. To avoid confusion here, we do not allow the user to directly modify the priors on the working parameters λ_1, λ_2, and ψ_D under the `"fa"` approach. The priors chosen above are approximately noninformative for most applications, and the user can further modify the exported **JAGS** code if he/she desires. Along with the prior distribution issue, the working model parameters are not identified by the likelihood because each inferential covariance parameter is the product of the three working parameters. This can complicate Laplace approximation of the marginal likelihood (which, as described below, is related to the Bayes factor), because the approximation works best when the posterior distributions are unimodal.

In **blavaan**, the user is free to specify `"srs"` or `"fa"` priors for all covariance parameters in the model. In the example section, we present a simple comparison of these two options' relative speeds and efficiencies. Beyond these options, the package attempts to identify when

it can sample latent variables directly from a multivariate normal distribution, which can improve sampling efficiency. This most commonly happens when we have many exogenous latent variables that all covary with one another. In this case, we place an inverse Wishart prior distribution on the latent variable covariance matrix.

While we prefer the `"srs"` approach to covariances between observed variables, the `"fa"` approach may be useful for complex models that run slowly. In these cases, it could be useful to sacrifice precise prior specification in favor of speed.

2.3. Model fit and comparison

Package **blavaan** supplies a variety of statistics for model evaluation and comparison, available via the `fitMeasures()` function. This function is illustrated later in the examples, and specific statistics are described in Appendix A. For model evaluation, **blavaan** supplies posterior predictive checks of the model's log-likelihood (e.g., Gelman, Carlin, Stern, and Rubin 2004). For model comparision, it supplies a variety of statistics including the Deviance Information Criterion (DIC; Spiegelhalter, Best, Carlin, and Van Der Linde 2002), the (Laplace-approximated) log-Bayes factor, the Widely Applicable Information Criterion (WAIC; Watanabe 2010), and the leave-one-out cross-validation statistic (LOO; e.g., Gelfand 1996). The latter two statistics are computed by R package **loo** (Vehtari, Gelman, and Gabry 2015), using output from **blavaan**.

Calculation of the information criteria is somewhat complicated by the fact that, during **JAGS** model estimation, we condition on the latent variables $\boldsymbol{\eta}_i$, $i = 1, \ldots, n$. That is, our model utilizes likelihoods of the form $L(\boldsymbol{\vartheta}, \boldsymbol{\eta}_i | \boldsymbol{y}_i)$, where $\boldsymbol{\vartheta}$ is the vector of model parameters and $\boldsymbol{\eta}_i$ is a vector of latent variables associated with individual i. Because the latent variables are random effects, we must integrate them out to obtain $L(\boldsymbol{\vartheta} | \boldsymbol{Y})$, the likelihood by which model assessment statistics are typically computed. This is easy to do for the models considered here, because the integrated likelihood continues to be multivariate normal. However, we cannot rely on **JAGS** to automatically calculate the correct likelihood, so that **blavaan** calculates the likelihood separately after parameters have been sampled in **JAGS**.

Now that we have provided background information on the models implemented in **blavaan**, the next section further describes how the package works. We then provide a series of examples.

3. Overview of blavaan

Readers familiar with **lavaan** will also be familiar with **blavaan**. The main functions are the same as the main **lavaan** functions, except that they start with the letter 'b'. For example, confirmatory factor analysis models are estimated via `bcfa()` and structural equation models are estimated via `bsem()`. These two functions call the more general `blavaan()` function with a specific arrangement of arguments. The **blavaan** model specification syntax is nearly the same as the **lavaan** model specification syntax, and many functions are used by both packages.

As compared to **lavaan**, there are a small number of new features in **blavaan** that are specific to Bayesian modeling: prior distribution specification, export/use of **JAGS** syntax, convergence diagnostics, and specification of initial values. We discuss these topics in the context of a simple, one-factor confirmatory model. The model uses the Holzinger and Swineford (1939) data included with **lavaan**, with code for model specification and estimation appearing below.

```
R> model <- ' visual =~ x1 + x2 + x3 '
R> fit <- bcfa(model, data = HolzingerSwineford1939, jagfile = TRUE)
```

3.1. Prior distribution specification

Package **blavaan** defines a default prior distribution for each type of model parameter (see Equations 1 and 2). These default priors can be viewed via

```
R> dpriors()
```

```
             nu          alpha         lambda           beta         itheta
"dnorm(0,1e-3)" "dnorm(0,1e-2)" "dnorm(0,1e-2)" "dnorm(0,1e-2)"  "dgamma(1,.5)"
           ipsi            rho           ibpsi            tau          delta
 "dgamma(1,.5)"    "dbeta(1,1)" "dwish(iden,3)"   "dnorm(0,.1)"  "dgamma(1,.5)"
```

which includes a separate default prior for ten types of parameters. Default prior distributions are placed on precisions instead of variances, so that the letter i in `itheta` and `ipsi` stands for "inverse." Most of these parameters correspond to notation from Equations 1 to 4, with the exception of `rho` and `ibpsi`. The `rho` prior is used only for the `"srs"` option, as it is for correlation parameters associated with covariances in Θ or Ψ. The `ibpsi` prior is used for the covariance matrix of latent variables that are all allowed to covary with one another (commonly, blocks of exogenous latent variables). This block prior can improve sampling efficiency and reduce autocorrelation.

For most parameters, the user is free to declare prior distributions using any prior distribution that is defined in **JAGS**. Changes to default priors can be supplied with the `dp` argument. Further, the modifiers `[sd]` and `[var]` can be used to put a prior on a standard deviation or variance parameter, instead of the corresponding precision parameter. For example,

```
R> fit <- bcfa(model, data = HolzingerSwineford1939,
+    dp = dpriors(nu = "dnorm(4,1)", itheta = "dunif(0,20)[sd]"))
```

sets the default priors for (i) the manifest variables' intercepts to normal with mean 4 and precision 1, and (ii) the manifest variables' standard deviations to uniform with bounds of 0 and 20.

Priors associated with the `rho` and `ibpsi` parameters are less amenable to change. The default `rho` prior is a beta(1,1) distribution with support on $(-1, 1)$; beta distributions with other parameter values could be used, but this distribution must be beta for now. The default prior on blocks of precision parameters is the Wishart with identity scale matrix and degrees of freedom equal to the dimension of the scale matrix plus one. The user cannot easily make changes to this prior distribution via the `dp` argument. Changes can be made, however, by exporting the **JAGS** syntax and manually editing it. This is further described in the next section.

In addition to priors on classes of model parameters, the user may wish to set the prior of a specific model parameter. This is accomplished by using the `prior()` modifier within the model specification. For example,

```
R> model <- ' visual =~ x1 + prior("dnorm(1,1)")*x2 + x3 '
```

sets a specific prior distribution for the loading parameter from the visual factor to x2. Priors set in this manner (via the model syntax) take precedent over the default priors set via the `dp` argument. Additionally, if one specifies a covariance parameter in the model syntax, the `prior()` modifier is for the correlation associated with the covariance, as opposed to the covariance itself. The prior should be a distribution with support on $(0, 1)$ (typically the beta distribution), which is automatically converted to an analogous distribution that has support on $(-1, 1)$.

3.2. JAGS syntax

Users may find that some desired features are not currently implemented in **blavaan**; such features may be related to, e.g., the use of specific types of priors or the handling of discrete variables. In these situations, we allow the user to export the **JAGS** syntax so that they can implement new features themselves. We believe this will be especially useful for researchers who wish to develop new types of models: these researchers can specify a traditional model in **blavaan** that is related to the one that they want to implement, and they can then obtain the **JAGS** syntax and data for the traditional model. This **JAGS** syntax should ease implementation of the novel model.

To export the **JAGS** syntax, users can employ the `jagfile` argument that was used in the `bcfa` command above. When `jagfile` is set to `TRUE`, the syntax will be written to the `lavExport` folder; if a character string is supplied, **blavaan** will use the character string as the folder name.

When the syntax is exported, two files are written within the folder. The first, `sem.jag`, contains the model specification in **JAGS** syntax. The second, `semjags.rda`, is a list named `jagtrans` that contains the data and initial values in **JAGS** format, as well as labels associated with model parameters. These pieces can be used to run the model "manually" via, e.g., package **rjags** (Plummer 2016) or **runjags** (Denwood 2016). The example below illustrates how to load the **JAGS** data (along with initial values and parameter labels) and estimate the exported model via **runjags**.

```
R> load("lavExport/semjags.rda")
R> fit <- run.jags("lavExport/sem.jag", monitor = jagtrans$coefvec$jlabel,
+    data = jagtrans$data, inits = jagtrans$inits)
```

If the user modifies the **JAGS** file, then the `monitor` and `inits` arguments might require modification (or, for automatic initial values from **JAGS**, the user could omit the `inits` argument). The data should not require modification unless the user adds or removes observed variables from the model.

We also note that the dimensions of the parameter vectors/matrices in the **JAGS** syntax generally match the dimensions that we would expect from Equations 1 and 2, with a third dimension being added for multiple group models. These matrix dimensions (and the specific nonzero entries within each matrix) are obtained directly from **lavaan**.

3.3. Convergence diagnostics

Package **blavaan** offers two methods for monitoring chain convergence, specified as the `"auto"` or `"manual"` options to the `convergence` argument. Regardless of which option is used, the default number of chains is three and can be modified via the `n.chains` argument.

Under `"auto"` convergence, chains are sampled for an initial period in order to (i) achieve convergence as determined by the potential scale reduction factor (PSRF; Gelman and Rubin 1992) and (ii) determine the number of samples necessary to obtain precise posterior estimates. Following this initial period, the chains are further sampled for the determined number of iterations. This automatic assessment comes directly from the `autorun.jags()` function of package **runjags** (Denwood 2016); see there for further detail.

Under `"manual"` convergence, the user can specify the desired number of adaptation, burnin, and sample iterations via arguments of the same name (with defaults of 1,000 adaptation iterations, 4,000 burnin iterations, and 10,000 sample iterations). Adaptation iterations are those where samplers can modify themselves to increase sampling efficiency, whereas burnin iterations are discarded samples obtained from "fixed" samplers (e.g., Plummer 2015). A warning is issued if any PSRF is greater than 1.2, though some users may desire a more stringent criterion that is closer to 1.0. Access to the parameters' PSRF values is obtained via

```
R> blavInspect(fit, "psrf")
```

Additionally, there is a `plot` method that includes the familiar time series plots, autocorrelation plots, and more. For example, autocorrelation plots for the first four free parameters are obtained via

```
R> plot(fit, 1:4, "autocorr")
```

where parameter numbers correspond to the ordering of `coef(fit)`. This plot functionality comes from package **runjags**, with plotting options being found in the help file for `plot.runjags()`.

3.4. Initial values

In many Bayesian SEMs, poorly-chosen initial values can lead to extremely slow convergence. Package **blavaan** provides multiple options for supplying initial values that vary from chain to chain and/or that improve chain convergence. These are obtained via the `inits` argument. Under option `"prior"` (default), starting values are random draws from each parameter's prior distribution. However, to aid in convergence, loading and regression parameters are all required to be positive and close to 1, and correlation parameters are required to be close to zero. Under option `"jags"`, starting values are automatically set by **JAGS**.

Additional options for starting values are obtained as byproducts of **lavaan**. For example, the options `"Mplus"` and `"simple"` are analogous to the options for the **lavaan** `start` argument, where the same initial values are used for all chains. Individual initial values can also be set via the `start()` modifier to the lavaan syntax, and these initial values will be used for all chains.

Finally, it is also possible to supply a full set of user-defined starting values for each chain. This is most easily accomplished by estimating the model once in order to obtain the list of initial values. The initial values of a fitted model can be obtained via `blavInspect()`; i.e.,

```
R> myinits <- blavInspect(fit, "inits")
```

The object `myinits` is a list whose length equals `n.chains`, with each entry being another list that contains initial values for the model's parameter vector. It may be helpful to call `blavInspect()` with the `"jagnames"` argument, in order to examine correspondence between **blavaan** parameter names and **JAGS** parameter names. Finally, after editing, a call to **blavaan** with the argument `inits = myinits` will estimate a model using the custom starting values.

Now that we have seen some features that are novel to **blavaan**, we will further illustrate the package by example.

4. Applications

In the applications below, we first illustrate some general features involving prior distribution specification and model fit measures. We then illustrate the study of measurement invariance via Bayesian models, which involves multiple across-group parameter constraints. Finally, we provide some advanced examples involving the direct modification of exported **JAGS** code and the use of informative prior distributions.

4.1. Political democracy

We begin with the Bollen (1989) industrialization-democracy model discussed earlier in the paper. The model specification is identical to the specification that we would use in **lavaan**:

```
R> model <- '
+        # latent variable definitions
+          ind60 =~ x1 + x2 + x3
+          dem60 =~ y1 + a*y2 + b*y3 + c*y4
+          dem65 =~ y5 + a*y6 + b*y7 + c*y8
+
+        # regressions
+          dem60 ~ ind60
+          dem65 ~ ind60 + dem60
+
+        # residual correlations
+          y1 ~~ y5
+          y2 ~~ y4 + y6
+          y3 ~~ y7
+          y4 ~~ y8
+          y6 ~~ y8
+ '
R> fit <- bsem(model, data = PoliticalDemocracy,
+    dp = dpriors(nu = "dnorm(5,1e-2)", itheta = "dlnorm(1,.1)[sd]",
+      ipsi = "dlnorm(1,.1)[sd]", rho = "dbeta(3,3)"),
+    jagcontrol = list(method = "rjparallel"))
```

where the latter argument, `jagcontrol`, allows us to sample each chain in parallel for faster model estimation. Defaults for the number of adaptation, burnin, and drawn samples come from package **runjags**; those defaults are currently 1,000, 4,000, and 10,000, respectively. Users can modify any of these via the arguments `adapt`, `burnin`, and `sample`. Trace plots of
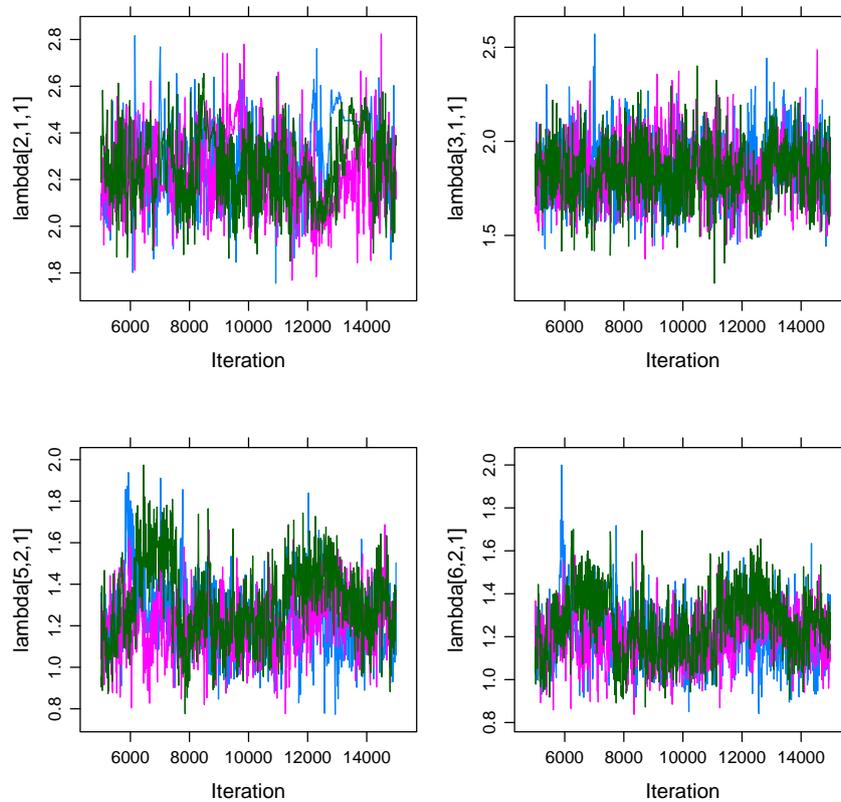
Figure 3: Trace plots of the first four parameters from the political democracy model.

sampled parameters can be obtained immediately; for example, the command below provides trace plots for the first four free parameters, with the resulting plot displayed in Figure 3.

```
R> plot(fit, 1:4, "trace")
```

As described earlier in the paper, users can also control the handling of residual covariances via the `cp` argument. For the particular model considered here, we compared the sampling speed and efficiency (excluding posterior predictive checks) of the `"srs"` and `"fa"` options using a four-core Dell Precision laptop running Ubuntu linux. We observed that the `"srs"` option took 76 seconds and the `"fa"` option took 65 seconds, illustrating the speed advantage of the `"fa"` approach. However, the ratio of effective sample sizes for the two approaches averaged 1.49 in favor of the `"srs"` approach (where the average is taken across parameters), implying that `"srs"` leads to more efficient sampling. In our experience, these results can vary depending on the specific model of interest, but it generally appears that `"fa"` is somewhat faster and `"srs"` somewhat more efficient.

Once we have sampled for the desired number of iterations, we can make use of **lavaan** functions in order to summarize and study the fitted model. Users may primarily be interested in `summary()`, which organizes results in a manner similar to the classical **lavaan** models.

```
R> summary(fit)
```

```
blavaan (0.3-2) results of 10000 samples after 5000 adapt/burnin iterations

  Number of observations                          75

  Number of missing patterns                       1

  Statistic                            MargLogLik         PPP
  Value                                 -1658.937        0.525

Parameter Estimates:


Latent Variables:
                  Estimate  Post.SD  HPD.025  HPD.975     PSRF     Prior
  ind60 =~
    x1               1.000
    x2               2.253    0.158    1.966    2.553    1.022   dnorm(0,1e-2)
    x3               1.850    0.166    1.526    2.176    1.004   dnorm(0,1e-2)
  dem60 =~
    y1               1.000
    y2         (a)   1.253    0.176    0.919    1.605    1.079   dnorm(0,1e-2)
    y3         (b)   1.221    0.144     0.94    1.504    1.069   dnorm(0,1e-2)
    y4         (c)   1.321    0.158    1.028    1.634    1.092   dnorm(0,1e-2)
  dem65 =~
    y5               1.000
    y6         (a)   1.253    0.176    0.919    1.605    1.079
    y7         (b)   1.221    0.144     0.94    1.504    1.069
    y8         (c)   1.321    0.158    1.028    1.634    1.092

Regressions:
                  Estimate  Post.SD  HPD.025  HPD.975     PSRF     Prior
  dem60 ~
    ind60            1.418    0.408    0.624    2.227    1.004   dnorm(0,1e-2)
  dem65 ~
    ind60            0.540    0.262    0.031     1.06    1.121   dnorm(0,1e-2)
    dem60            0.883    0.082    0.746    1.059    1.018   dnorm(0,1e-2)

Covariances:
                  Estimate  Post.SD  HPD.025  HPD.975     PSRF     Prior
 .y1 ~~
   .y5               0.645    0.381   -0.053     1.41    1.012     dbeta(3,3)
 .y2 ~~
   .y4               1.486    0.728    0.136    2.961    1.002     dbeta(3,3)
   .y6               2.204    0.742    0.776     3.61    1.003     dbeta(3,3)
 .y3 ~~
   .y7               0.796    0.640    -0.47     2.06    1.013     dbeta(3,3)
 .y4 ~~
   .y8               0.303    0.496   -0.636    1.277    1.007     dbeta(3,3)
 .y6 ~~
   .y8               1.248    0.583     0.15    2.402    1.002     dbeta(3,3)

Intercepts:
                  Estimate  Post.SD  HPD.025  HPD.975     PSRF     Prior
   .x1              5.059    0.078    4.905    5.216    1.001   dnorm(5,1e-2)
   .x2              4.804    0.160    4.497    5.144    1.001   dnorm(5,1e-2)
   .x3              3.567    0.152    3.266    3.871    1.000   dnorm(5,1e-2)
   .y1              5.480    0.290    4.905     6.05    1.020   dnorm(5,1e-2)
```

```
 .y2                4.279    0.436    3.419    5.106    1.011    dnorm(5,1e-2)
 .y3                6.586    0.393    5.827    7.375    1.016    dnorm(5,1e-2)
 .y4                4.475    0.371    3.714    5.173    1.022    dnorm(5,1e-2)
 .y5                5.154    0.295    4.551    5.723    1.016    dnorm(5,1e-2)
 .y6                3.011    0.380    2.251    3.734    1.010    dnorm(5,1e-2)
 .y7                6.219    0.356     5.52    6.917    1.017    dnorm(5,1e-2)
 .y8                4.073    0.362    3.362    4.781    1.016    dnorm(5,1e-2)
  ind60             0.000
 .dem60             0.000
 .dem65             0.000

Variances:
                 Estimate  Post.SD  HPD.025  HPD.975     PSRF    Prior
 .x1                0.094    0.023    0.048    0.138    1.012 dlnorm(1,.1)[sd]
 .x2                0.096    0.077        0    0.234    1.034 dlnorm(1,.1)[sd]
 .x3                0.521    0.103    0.331    0.729    1.008 dlnorm(1,.1)[sd]
 .y1                2.105    0.529    1.113    3.145    1.018 dlnorm(1,.1)[sd]
 .y2                8.113    1.509    5.314   11.069    1.001 dlnorm(1,.1)[sd]
 .y3                5.464    1.090    3.441     7.59    1.011 dlnorm(1,.1)[sd]
 .y4                3.407    0.820     1.89    5.048    1.003 dlnorm(1,.1)[sd]
 .y5                2.584    0.549     1.59    3.704    1.027 dlnorm(1,.1)[sd]
 .y6                5.166    0.971    3.398    7.127    1.002 dlnorm(1,.1)[sd]
 .y7                3.866    0.798    2.422    5.493    1.005 dlnorm(1,.1)[sd]
 .y8                3.403    0.783    1.941    4.963    1.004 dlnorm(1,.1)[sd]
  ind60             0.449    0.093    0.284    0.632    1.005 dlnorm(1,.1)[sd]
 .dem60             3.812    1.003    1.938    5.732    1.058 dlnorm(1,.1)[sd]
 .dem65             0.152    0.198        0    0.543    1.196 dlnorm(1,.1)[sd]
```

While the results look similar to those that one would see in **lavaan**, there are a few differences that require explanation. First, the top of the output includes two model evaluation measures: a Laplace approximation of the marginal log-likelihood and the posterior predictive p value (see Appendix A). Second, the "Parameter Estimates" section contains many new columns. These are the posterior mean (`Post.Mean`), the posterior standard deviation (`Post.SD`), a 95% highest posterior density interval (`HPD.025` and `HPD.975`), the potential scale reduction factor for assessing chain convergence (`PSRF`; Gelman and Rubin 1992), and the prior distribution associated with each model parameter (`Prior`). Users can optionally obtain posterior medians, posterior modes, and effective sample sizes (number of posterior samples drawn, adjusted for autocorrelation). These can be obtained by supplying the logical arguments `postmedian`, `postmode`, and `neff` to `summary()`.

Finally, we can obtain various Bayesian model assessment measures via `fitMeasures()`

```
R> fitMeasures(fit)


      npar         logl          ppp          bic          dic        p_dic         waic
    39.000    -1550.509        0.525     3268.876     3174.329       36.656     3179.278
    p_waic        looic        p_loo margloglik
    38.836     3179.962       39.178  -1658.937
```

where, as previously mentioned, the WAIC and LOO statistics are computed with the help of package **loo**. Other **lavaan** functions, including `parameterEstimates()` and `parTable()`, similarly work as expected.

4.2. Measurement invariance

Verhagen and Fox (2013; see also Verhagen, Levy, Millsap, and Fox 2016) describe the use of Bayesian methods for studying the measurement invariance of sets of items or scales. Briefly, measurement invariance means that the items or scales measure diverse groups of individuals in a fair manner. For example, two individuals with the same level of mathematics ability should receive the same score on a mathematics test (within random noise), regardless of the individuals' backgrounds and demographic variables. In this section, we illustrate a Bayesian measurement invariance approach using the Holzinger and Swineford (1939) data. This section also illustrates how we can estimate a model in **blavaan** using pieces of a fitted **lavaan** object. This may be of interest to advanced users who would prefer to write/edit a lavaan "parameter table" instead of using model specification syntax. Additionally, the use of a **lavaan** object provides a path from **Mplus** to **blavaan**, via the `mplus2lavaan()` function from **lavaan**.

We first fit three increasingly-restricted models to the Holzinger and Swineford (1939) data via classical methods. This is accomplished via the **lavaan** `cfa()` function (though we could also immediately use `bcfa()` here):

```
R> HS.model <- ' visual  =~ x1 + x2 + x3
+                textual =~ x4 + x5 + x6
+                speed   =~ x7 + x8 + x9 '
R> fit1 <- cfa(HS.model, data = HolzingerSwineford1939, group = "school")
R> fit2 <- cfa(HS.model, data = HolzingerSwineford1939, group = "school",
+    group.equal = "loadings")
R> fit3 <- cfa(HS.model, data = HolzingerSwineford1939, group = "school",
+    group.equal = c("loadings", "intercepts"))
```

The resulting objects each include a "parameter table," which contains all model details necessary for **lavaan** estimation. To fit these models via Bayesian methods, we can merely pass the classical model's parameter table and data to **blavaan**. This is accomplished via

```
R> bfit1 <- bcfa(parTable(fit1), data = HolzingerSwineford1939,
+    group = "school")
R> bfit2 <- bcfa(parTable(fit2), data = HolzingerSwineford1939,
+    group = "school")
R> bfit3 <- bcfa(parTable(fit3), data = HolzingerSwineford1939,
+    group = "school")
```

As mentioned above, we could also supply the `group.equal` argument directly to the `bcfa()` calls.

Following model estimation, we can immediately compare the models via `fitMeasures()`. For example, focusing on the first two models, we obtain

```
R> fitMeasures(bfit1)

    npar        logl         ppp         bic         dic       p_dic        waic
  60.000   -3684.461       0.000    7710.948    7483.196      57.137    7492.991
  p_waic       looic       p_loo margloglik
  64.597    7493.319      64.761   -3937.468
```

```
R> fitMeasures(bfit2)
```

```
     npar        logl         ppp         bic         dic       p_dic        waic
   54.000   -3687.297       0.000    7682.417    7478.802      52.104    7485.929
   p_waic       looic       p_loo  margloglik
   57.184    7486.290      57.365   -3917.959
```

As judged by the posterior predictive p value (`ppp`), neither of the two models provide a good fit to the data. In practice, we might stop our measurement invariance study there. However, for the purpose of demonstration, we can also examine the information criteria. Based on these, we would prefer the second model (due to the smaller DIC, WAIC, and LOOIC values).

4.3. Informative prior distributions

We have seen that **blavaan** allows users to specify prior distributions for individual model parameters and for classes of model parameters. In this section, we illustrate how informative prior distributions can be set for factor analysis parameters in the context of a specific dataset, where priors are set based on the parameters' interpretations.

We consider a one-factor, three-indicator model fit to a subset of data from a stereotype threat study (Wicherts, Dolan, and Hessen 2005). The data are available in R package **psychotools** (Zeileis, Strobl, Wickelmaier, Komboz, and Kopf 2018), and the model specifies a general "ability" factor underlying three academic tests (of verbal ability, numerical ability, and abstract reasoning). The test scores are sums of the number of items answered correctly, so they are bounded from below at 0 and from above at 16 (verbal), 14 (numerical), and 18 (abstract reasoning). While these bounds technically violate the normality assumption of the factor analysis model, it is customary to apply this model to test scores (especially in the current situation, where the test scores are unimodal with modes in the middle of the score ranges). A factor analysis model with default (non-informative) prior distributions and automatic convergence assessment can be fit to the data via

```
R> library("psychotools")
R> data("StereotypeThreat")
R> st <- subset(StereotypeThreat, ethnicity == "majority")
R> model <- ' ability =~ abstract + verbal + numerical '
R> bfit <- bcfa(model, data = st, convergence = "auto")
```

Instead of using the default prior distributions, we can tailor prior distributions to this particular dataset by simultaneously considering the data and each model parameter's interpretation. First, because the test scores are all bounded, we know the maximum possible standard deviation on each test (obtained if half the participants scored a 0 and half score the maximum). These are 8 for the verbal test, 7 for the numerical test, and 9 for the abstract reasoning test. Thus, we place uniform priors on the manifest variables' standard deviations with lower bounds at 0 and upper bounds at these maxima:

```
R> model <- c(model,
+    ' abstract ~~ prior("dunif(0,9)[sd]")*abstract
+      verbal ~~ prior("dunif(0,8)[sd]")*verbal
+      numerical ~~ prior("dunif(0,7)[sd]")*numerical ')
```

		Parameter								
Prior		λ_2	λ_3	θ_1	θ_2	θ_3	ψ_1	ν_1	ν_2	ν_3
Default	Est	1.1	1.58	7.9	8.43	1.49	2.06	9.84	6.97	5.43
	SD	0.33	0.49	1.11	1.2	1.04	0.94	0.25	0.26	0.19
Inform.	Est	1.06	1.37	7.73	8.39	2.03	2.51	9.83	6.99	5.44
	SD	0.33	0.48	1.24	1.23	1.13	1.18	0.25	0.26	0.19

Table 1: Comparison of posterior means and standard deviations under default prior distributions and informative prior distributions.

Next, we consider the factor variance. Because the loading associated with the abstract reasoning test will be fixed at 1, the factor variance can be interpreted as the variability in the abstract reasoning test that can be attributed to the ability factor. Because the three tests were chosen to be highly correlated (and because the total standard deviation of abstract reasoning is 9 or less), we use a uniform prior with an upper bound of 6 here. Thus, even in the case where we observe the maximum standard deviation of 9, the factor could still account for 2/3 of the variability in the ability test.

```
R> model <- c(model, ' ability ~~ prior("dunif(0,4.5)[sd]")*ability ')
```

Next, we consider the manifest variables' intercept parameters, which reflect average scores on each of the three tests. These tests are known historically to result in average scores near the middle of the range (Wicherts *et al.* 2005), so our prior distributions are truncated normals centered at the middle of each test's range (with the truncation points at the minimum and maximum possible scores on each test):

```
R> model <- c(model,
+    ' abstract ~ prior("dnorm(9,.25) T(0,18)")*1
+      verbal ~ prior("dnorm(8,.25) T(0,16)")*1
+      numerical ~ prior("dnorm(7,.25) T(0,14)")*1 ')
```

Finally, we consider the factor loadings. We have no knowledge that the ability factor will influence one test more than others, so we might expect the two free loadings to be around 1 (because the fixed loading equals 1). We also expect positive loadings, because ability should positively influence all three tests. Thus, we supply uniform(0,3) priors to the `dp` argument of `bcfa()`:

```
R> bfitInform <- bcfa(model, data = st,
+    dp = dpriors(lambda = "dunif(0,3)"), convergence = "auto")
```

The two models' fit measures are generally similar and not shown. We compare the two models' posterior means and standard deviations in Table 1. The differences are unlikely to impact substantive conclusions, but two of them are noteworthy. First, the factor variance ψ_1 is larger under the model with informative priors, likely because the informative prior (uniform(0,4.5) on the standard deviation) placed more density on larger values of the standard deviation. We observe a similar phenomenon with the residual variance of the numerical test (θ_3). Second, the posterior means of the loadings (λ_2 and λ_3) are somewhat smaller under the informative priors. This is likely related to the larger factor variance estimates.

```
model {
  for(i in 1:N) {
    abstract[i] ~ dt(mu[i,1], 1/theta[1,1,g[i]], df)
    verbal[i] ~ dt(mu[i,2], 1/theta[2,2,g[i]], df)
    numerical[i] ~ dt(mu[i,3], 1/theta[3,3,g[i]], df)

    # lvs
    eta[i,1] ~ dt(mu_eta[i,1], 1/psi[1,1,g[i]], df)
  }

  df <- 1/dfinv
  dfinv ~ dunif(1/200, 1)

  # mu definitions
```

Figure 4: Illustration of **JAGS** modifications necessary to implement the Zhang *et al.* (2014) robust factor analysis model. The original **JAGS** code exported from **blavaan** is in black font, with modifications and additions in green. Additional code requiring no modification (extending below the "`mu definitions`" comment) is omitted.

This example could be used as the start of a larger analysis of posterior sensitivity to prior distributions. For the purpose of automation, prior distributions could be entered directly into the model's parameter table (obtained via, e.g., `parTable(bfit)`) and the model subsequently re-estimated, similar to what was done in the measurement invariance example.

4.4. Extensions of JAGS syntax

Finally, we provide an example involving use of the **JAGS** syntax to estimate novel models. Consider the robust factor analysis model of Zhang, Li, and Liu (2014), which essentially replaces the factor analysis model's normal distributions with t distributions. In particular, each factor η arises from a $t_{\mathrm{df}}(0,1)$ distribution, and each residual ϵ_j arises from a $t_{\mathrm{df}}(0,\psi_j)$ distribution. The degrees of freedom, df, is a free parameter and is shared by all the t distributions. To our knowledge, there exists no software to readily estimate this model, and Zhang *et al.* (2014) implemented their own Gibbs sampler (making use of analytic posterior distributions under conjugate priors). Here, we show how the **JAGS** syntax from **blavaan** can be modified to easily estimate this model. For illustration, we again apply a one-factor model to the Wicherts *et al.* (2005) data from the previous section.

We begin by using **blavaan** to export **JAGS** syntax for a simple, one-factor model.

```
R> model <- ' ability =~ abstract + verbal + numerical '
R> bfit <- bcfa(model, data = st, jagfile = TRUE)
```

By default, the exported **JAGS** code is written to the file `lavExport/sem.jag`. Upon opening that file, we must make two edits (see Figure 4). First, the normal distributions (`dnorm()`) are replaced with t distributions (`dt()`). Second, we need to specify a prior distribution for the degrees of freedom parameter. Similar to Zhang *et al.* (2014), we place a flat prior on the inverse degrees of freedom. The lower bound is nonzero to aid in convergence; this is justified

by realizing that, as the degrees of freedom increase, the t distribution becomes the normal distribution. Once we surpass, say, 200 degrees of freedom, we can accept that larger values are practically equivalent to 200.

The modified syntax from Figure 4 could then be estimated manually, making use of the exported data and adding a monitor for the `df` parameter. For example, if the exported files are saved as `lavExport/robustfa.jag` and `lavExport/robustfa.rda`, the model can be estimated via

```
R> load("lavExport/robustfa.rda")
R> fit <- run.jags("lavExport/robustfa.jag",
+    monitor = c(jagtrans$monitors, "df"), data = jagtrans$data,
+    inits = jagtrans$inits)
```

While many model extensions will be more complicated than the one considered here, this example illustrates **blavaan**'s potential use for statistical researchers who are developing new models: instead of writing **JAGS** syntax entirely from scratch, these researchers may use **blavaan** to obtain syntax for a basic model that is similar to the desired model. In many cases, this will simplify implementation of the desired model.

5. Conclusion

As described throughout the paper, package **blavaan** combines many existing tools so that it is easy to estimate multivariate normal SEMs via open-source software. The package can be useful to applied researchers who need an expanded set of prior distributions at their disposal, freeing them from the need to learn the intricacies of an MCMC package. It can also be useful to methodological researchers, freeing them from the need to code their own **JAGS** models from scratch. The idea of separating model specification from MCMC coding seems generally useful beyond SEM, and others are indeed making progress in this direction. For example, package **rstanarm** (Stan Development Team 2018) allows users to estimate generalized linear mixed models via Stan, using **lme4** (Bates, Mächler, Bolker, and Walker 2015) syntax. The coding of complex models in BUGS, **JAGS**, or Stan is often tedious even for experienced users, so that progress here could improve most users' workflows.

There are a variety of additional models that we plan to support in future versions of **blavaan**, including models with latent interactions, with ordinal variables, and with mixture and multi-level components. These models have been addressed in the literature (e.g., Asparouhov and Muthén 2010; Song and Lee 2001, 2012), and **JAGS** examples for estimating specific instances of these models is available (e.g., Cho, Preacher, and Bottge 2015; Merkle and Wang 2018). Work remains, however, to sample from these models efficiently and to specify the models via **lavaan**'s syntax. The sampling efficiency of Stan may be useful for at least some of these models, and we plan to explore this in the future.

In summary, **blavaan** is currently useful for estimating many types of multivariate normal SEMs, and the **JAGS** export feature allows researchers to extend the models in any fashion desired. As additional features are added, we hope that the package keeps pace with **lavaan** as an open set of tools for SEM estimation and study.

Acknowledgments

References

Alvarez I, Niemi J, Simpson M (2014). "Bayesian Inference for a Covariance Matrix." In *Proceedings of the 26th Annual Conference on Applied Statistics in Agriculture*, pp. 71–82.

Asparouhov T, Muthén B (2010). *Bayesian Analysis Using **Mplus**: Technical Implementation.* Version 3, URL http://www.statmodel.com/download/Bayes3.pdf.

Barnard J, McCulloch R, Meng XL (2000). "Modeling Covariance Matrices in Terms of Standard Deviations and Correlations, with Application to Shrinkage." *Statistica Sinica*, **10**, 1281–1311. doi:10.5705/ss.2010.253a.

Bates D, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using **lme4**." *Journal of Statistical Software*, **67**(1), 1–48. doi:10.18637/jss.v067.i01.

Beaujean AA (2014). *Latent Variable Modeling Using R.* Routledge, New York.

Bollen KA (1989). *Structural Equations with Latent Variables.* John Wiley & Sons. doi:10.1002/9781118619179.

Carpenter B, Gelman A, Hoffman MD, Lee D, Goodrich B, Betancourt M, Brubaker M, Guo J, Li P, Riddell A (2017). "Stan: A Probabilistic Programming Language." *Journal of Statistical Software*, **76**(1), 1–32. doi:10.18637/jss.v076.i01.

Chib S, Greenberg E (1998). "Analysis of Multivariate Probit Models." *Biometrika*, **85**, 347–361. doi:10.1093/biomet/85.2.347.

Cho SJ, Preacher KJ, Bottge BA (2015). "Detecting Intervention Effects in a Cluster-Randomized Design Using Multilevel Structural Equation Modeling for Binary Responses." *Applied Psychological Measurement*, **39**, 627–642. doi:10.1177/0146621615591094.

Cole DA, Ciesla JA, Steiger JH (2007). "The Insidious Effects of Failing to Include Design-Driven Correlated Residuals in Latent-Variable Covariance Structure Analysis." *Psychological Methods*, **12**, 381–398. doi:10.1037/1082-989x.12.4.381.

Denwood MJ (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:10.18637/jss.v071.i09.

Gelfand AE (1996). "Model Determination Using Sampling-Based Methods." In WR Gilks, S Richardson, DJ Spiegelhalter (eds.), *Markov Chain Monte Carlo in Practice*, pp. 145–162. Chapman & Hall, London.

Gelman A (2004). "Parameterization and Bayesian Modeling." *Journal of the American Statistical Association*, **99**, 537–545. `doi:10.1198/016214504000000458`.

Gelman A (2006). "Prior Distributions for Variance Parameters in Hierarchical Models." *Bayesian Analysis*, **3**, 515–534. `doi:10.1214/06-ba117a`.

Gelman A, Carlin JB, Stern HS, Rubin DB (2004). *Bayesian Data Analysis.* 2nd edition. Chapman & Hall, London.

Gelman A, Rubin DB (1992). "Inference from Iterative Simulation Using Multiple Sequences." *Statistical Science*, **7**, 457–511. `doi:10.1214/ss/1177011136`.

Ghosh J, Dunson DB (2009). "Default Prior Distributions and Efficient Posterior Computation in Bayesian Factor Analysis." *Journal of Computational and Graphical Statistics*, **18**, 306–320. `doi:10.1198/jcgs.2009.07145`.

Hoijtink H, Van de Schoot R (2018). "Testing Small Variance Priors Using Prior-Posterior Predictive P-Values." *Psychological Methods*. `doi:10.1037/met0000131`. Forthcoming.

Holzinger KJ, Swineford FA (1939). *A Study of Factor Analysis: The Stability of a Bi-Factor Solution.* Number 48 in Supplementary Educational Monograph. University of Chicago Press, Chicago.

Jöreskog KG, Sörbom D (1997). **LISREL** *8 User's Reference Guide.* Scientific Software International.

Kass RE, Raftery AE (1995). "Bayes Factors." *Journal of the American Statistical Association*, **90**, 773–795. `doi:10.2307/2291091`.

Lee SY (2007). *Structural Equation Modeling: A Bayesian Approach.* John Wiley & Sons.

Lewis SM, Raftery AE (1997). "Estimating Bayes Factors via Posterior Simulation with the Laplace-Metropolis Estimator." *Journal of the American Statistical Association*, **92**, 648–655. `doi:10.2307/2965712`.

Liu CC, Aitkin M (2008). "Bayes Factors: Prior Sensitivity and Model Generalizability." *Journal of Mathematical Psychology*, **52**, 362–375. `doi:10.1016/j.jmp.2008.03.002`.

Lu ZH, Chow SM, Loken E (2016). "Bayesian Factor Analysis as a Variable-Selection Problem: Alternative Priors and Consequences." *Multivariate Behavioral Research*, **51**, 519–539. `doi:10.1080/00273171.2016.1168279`.

Lunn D, Jackson C, Best N, Thomas A, Spiegelhalter D (2012). *The BUGS Book: A Practical Introduction to Bayesian Analysis.* Chapman & Hall/CRC, Boca Raton.

Lunn DJ, Thomas A, Best N, Spiegelhalter D (2000). "WinBUGS – A Bayesian Modelling Framework: Concepts, Structure, and Extensibility." *Statistics and Computing*, **10**, 325–337. `doi:10.1023/a:1008929526011`.

Marsh HW, Wen Z, Hau KT, Nagengast B (2013). "Structural Equation Models of Latent Interaction and Quadratic Effects." In GR Hancock, RO Mueller (eds.), *Structural Equation Modeling: A Second Course*, 2nd edition, pp. 267–308. Information Age Publishing.

Martin AD, Quinn KM, Park JH (2011). "**MCMCpack**: Markov Chain Monte Carlo in R." *Journal of Statistical Software*, **42**, 1–21. `doi:10.18637/jss.v042.i09`.

Martin JK, McDonald RP (1975). "Bayesian Estimation in Unrestricted Factor Analysis: A Treatment for Heywood Cases." *Psychometrika*, **40**, 505–517. `doi:10.1007/bf02291552`.

Merkle EC, Wang T (2018). "Bayesian Latent Variable Models for the Analysis of Experimental Psychology Data." *Psychonomic Bulletin & Review*, **25**, 256–270. `doi:10.3758/s13423-016-1016-7`.

Millar RB (2009). "Comparison of Hierarchical Bayesian Models for Overdispersed Count Data Using DIC and Bayes' Factors." *Biometrics*, **65**, 962–969. `doi:10.1111/j.1541-0420.2008.01162.x`.

Murray J (2014). **bfa***: Bayesian Factor Analysis*. R package version 0.3.1, URL `https://CRAN.R-project.org/package=bfa`.

Muthén B, Asparouhov T (2012). "Bayesian Structural Equation Modeling: A More Flexible Representation of Substantive Theory." *Psychological Methods*, **17**, 313–335. `doi:10.1037/a0026802`.

Palomo J, Dunson DB, Bollen K (2007). "Bayesian Structural Equation Modeling." In SY Lee (ed.), *Handbook of Latent Variable and Related Models*, pp. 163–188. Elsevier.

Plummer M (2003). "**JAGS**: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling." In K Hornik, F Leisch, A Zeileis (eds.), *Proceedings of the 3rd International Workshop on Distributed Statistical Computing (DSC 2003)*. Technische Universität Wien, Vienna, Austria. URL `https://www.R-project.org/conferences/DSC-2003/Proceedings/Plummer.pdf`.

Plummer M (2015). **JAGS** *Version 4.0.0 User Manual*.

Plummer M (2016). **rjags***: Bayesian Graphical Models Using MCMC*. R package version 4-6, URL `http://CRAN.R-project.org/package=rjags`.

Raftery AE (1993). "Bayesian Model Selection in Structural Equation Models." In KA Bollen, JS Long (eds.), *Testing Structural Equation Models*, pp. 163–180. Sage Publications, Beverly Hills.

Rindskopf D (1984). "Using Phantom and Imaginary Latent Variables to Parameterize Constraints in Linear Structural Models." *Psychometrika*, **49**, 37–47. `doi:10.1007/bf02294204`.

Rosseel Y (2012). "**lavaan**: An R Package for Structural Equation Modeling." *Journal of Statistical Software*, **48**(2), 1–36. `doi:10.18637/jss.v048.i02`.

Rouder JN, Morey RD, Verhagen J, Province JM, Wagenmakers EJ (2016). "Is There a Free Lunch in Inference?" *Topics in Cognitive Science*, **8**, 520–547. `doi:10.1111/tops.12214`.

Scheines R, Hoijtink H, Boomsma A (1999). "Bayesian Estimation and Testing of Structural Equation Models." *Psychometrika*, **64**, 37–52. `doi:10.1007/bf02294318`.

Song XY, Lee SY (2001). "Bayesian Estimation and Test for Factor Analysis Model with Continuous and Polytomous Data in Several Populations." *British Journal of Mathematical and Statistical Psychology*, **54**, 237–263. `doi:10.1348/000711001159546`.

Song XY, Lee SY (2012). *Basic and Advanced Bayesian Structural Equation Modeling: With Applications in the Medical and Behavioral Sciences.* John Wiley & Sons. `doi:10.1002/9781118358887`.

Spiegelhalter DJ, Best NG, Carlin BP, Van Der Linde A (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society B*, **64**, 583–639. `doi:10.1111/1467-9868.00353`.

Stan Development Team (2018). **rstanarm**: *Bayesian Applied Regression Modeling via Stan*. R package version 2.17.4, URL `https://CRAN.R-project.org/package=rstanarm`.

Vanpaemel W (2010). "Prior Sensitivity in Theory Testing: An Apologia for the Bayes Factor." *Journal of Mathematical Psychology*, **54**, 491–498. `doi:10.1016/j.jmp.2010.07.003`.

Vehtari A, Gelman A, Gabry J (2015). **loo**: *Efficient Leave-One-Out Cross-Validation and WAIC for Bayesian Models*. URL `https://github.com/jgabry/loo`.

Vehtari A, Gelman A, Gabry J (2016). "Efficient Implementation of Leave-One-Out Cross-Validation and WAIC for Evaluating Fitted Bayesian Models." arXiv:1507.04544 [stat.CO], URL `http://arxiv.org/abs/1507.04544`.

Verhagen AJ, Fox JP (2013). "Bayesian Tests of Measurement Invariance." *British Journal of Mathematical and Statistical Psychology*, **66**, 383–401. `doi:10.1111/j.2044-8317.2012.02059.x`.

Verhagen J, Levy R, Millsap RE, Fox JP (2016). "Evaluating Evidence for Invariant Items: A Bayes Factor Applied to Testing Measurement Invariance in IRT Models." *Journal of Mathematical Psychology*, **72**, 171–182. `doi:10.1016/j.jmp.2015.06.005`.

Watanabe S (2010). "Asymptotic Equivalence of Bayes Cross Validation and Widely Applicable Information Criterion in Singular Learning Theory." *Journal of Machine Learning Research*, **11**, 3571–3594. `doi:10.1109/scis-isis.2012.6505025`.

Wicherts JM, Dolan CV, Hessen DJ (2005). "Stereotype Threat and Group Differences in Test Performance: A Question of Measurement Invariance." *Journal of Personality and Social Psychology*, **89**(5), 696–716. `doi:10.1037/0022-3514.89.5.696`.

Zeileis A, Strobl C, Wickelmaier F, Komboz B, Kopf J (2018). **psychotools**: *Infrastructure for Psychometric Modeling*. R package version 0.4-3, URL `https://CRAN.R-project.org/package=psychotools`.

Zhang J, Li J, Liu C (2014). "Robust Factor Analysis Using the Multivariate *t*-Distribution." *Statistica Sinica*, **24**, 291–312. `doi:10.5705/ss.2012.022`.

A. Details on model assessment statistics

In the subsections below, we review the model evaluation and comparison metrics that **blavaan** supplies.

Posterior predictive checks

As a measure of the model's absolute fit, **blavaan** computes a posterior predictive p value that compares observed likelihood ratio test (χ^2) statistics to likelihood ratio test statistics generated from the model's posterior predictive distribution (also see Muthén and Asparouhov 2012; Scheines *et al.* 1999). The likelihood ratio test statistic is generally computed via

$$\mathrm{LRT}(\boldsymbol{Y}, \boldsymbol{\vartheta}) = -2 \log L(\boldsymbol{\vartheta}|\boldsymbol{Y}) + 2 \log L(\boldsymbol{\vartheta}^{\mathrm{sat}}|\boldsymbol{Y}),$$

where $\boldsymbol{\vartheta}^{\mathrm{sat}}$ is a "saturated" parameter vector that perfectly matches the observed mean and covariance matrix.

For each posterior draw $\boldsymbol{\vartheta}^s$ ($s = 1, \ldots, S$), computation of the posterior predictive p value includes four steps:

1. Compute the observed LRT statistic as $\mathrm{LRT}(\boldsymbol{Y}, \boldsymbol{\vartheta}^s)$.

2. Generate artificial data $\boldsymbol{Y}^{\mathrm{rep}}$ from the model, assuming parameter values equal to $\boldsymbol{\vartheta}^s$.

3. Compute the posterior predictive LRT statistic $\mathrm{LRT}(\boldsymbol{Y}^{\mathrm{rep}}, \boldsymbol{\vartheta}^s)$.

4. Record whether or not $\mathrm{LRT}(\boldsymbol{Y}^{\mathrm{rep}}, \boldsymbol{\vartheta}^s) > \mathrm{LRT}(\boldsymbol{Y}, \boldsymbol{\vartheta}^s)$.

The posterior predictive p value is then the proportion of the time (out of S draws) that the posterior predictive LRT statistic is larger than the observed LRT statistic. Values closer to 0.5 indicate a model that fits the observed data, while values closer to 0 indicate the opposite. For use in practice, Muthén and Asparouhov (2012) use a threshold of 0.05 (analogous to a frequentist α level) and present some evidence that, as compared to the frequentist likelihood ratio test, the posterior predictive p is less sensitive to minor model misspecifications and exhibits better performance at small sample sizes. On the other hand, Hoijtink and Van de Schoot (2018) show that the posterior predictive p value misbehaves in situations involving highly informative prior distributions (as are sometimes used for shrinkage/regularization). They recommend a prior-posterior predictive p value that may be implemented in future versions of **blavaan**.

In **blavaan**, the posterior predictive p value is currently impractical when there are missing data. This is because it is impractical to compute the saturated log-likelihood in the presence of missing data. We need to use, e.g., the EM algorithm to do this, and this needs to be done $(S + 1)$ times (once for the saturated likelihood of the observed data and once for the saturated likelihood of each of the S artificial datasets). With complete data, we can more simply calculate the saturated log-likelihood using the sample mean vector and covariance matrix. The argument `test = "none"` can be supplied to **blavaan** in order to bypass these slow calculations when there are missing data, and future versions may sample the missing observations in order to avoid the issue.

DIC

The DIC is given as

$$DIC = -2 \log L(\hat{\boldsymbol{\vartheta}}|\boldsymbol{Y}) + 2 \text{ efp}_{\text{DIC}}, \tag{7}$$

where $\hat{\boldsymbol{\vartheta}}$ is a vector of posterior parameter means (or other measure of central tendency), $L(\hat{\boldsymbol{\vartheta}}|\boldsymbol{Y})$ is the model's likelihood at the posterior means, and efp_{DIC} is the model's effective number of parameters, calculated as

$$\text{efp}_{\text{DIC}} = 2 \left[\log L(\hat{\boldsymbol{\vartheta}}|\boldsymbol{Y}) - \overline{\log L(\boldsymbol{\vartheta}|\boldsymbol{Y})} \right]. \tag{8}$$

The latter term, $\overline{\log L(\boldsymbol{\vartheta}|\boldsymbol{Y})}$, is obtained by calculating the log-likelihood for each posterior sample and then averaging.

Many readers will be familiar with the automatic calculation of DIC within programs like BUGS and **JAGS**. As we described above, the automatic DIC is not what users typically desire because the likelihood conditions on the latent variables $\boldsymbol{\eta}_i$. This greatly increases the effective number of parameters and may result in poor inferences (for further discussion of this issue, see Millar 2009). As previously mentioned, **blavaan** avoids the automatic DIC computation in **JAGS**, calculating its own likelihoods after model parameters have been sampled.

WAIC and LOO

The WAIC and LOO are asymptotically equivalent measures that are advantageous over DIC while also being more difficult to compute than DIC. Many computational difficulties are overcome in developments by Vehtari, Gelman, and Gabry (2016), with their methodology being implemented in package **loo** (Vehtari *et al.* 2015). As input, **loo** requires casewise log-likelihoods associated with a set of posterior samples: $\log L(\boldsymbol{\vartheta}^s|\boldsymbol{y}_i), s = 1, \ldots, S; i = 1, \ldots, n$.

Like DIC, both of these measures seek to characterize a model's predictive accuracy (generalizability). The definition of WAIC looks similar to that of DIC:

$$\text{WAIC} = -2 \text{ lppd} + 2 \text{ efp}_{\text{WAIC}},$$

where the first term is related to log-likelihoods of observed data and the second term involves an effective number of parameters. Both terms now involve log-likelihoods associated with individual observations, however, which help us estimate the model's expected log pointwise predictive density (a measure of predictive accuracy).

The first term, the log pointwise predictive density of the observed data (lppd), is estimated via

$$\text{lppd} = \sum_{i=1}^{n} \log \left(\frac{1}{S} \sum_{s=1}^{S} f(\boldsymbol{y}_i|\boldsymbol{\vartheta}^s) \right), \tag{9}$$

where S is the number of posterior draws and $f(\boldsymbol{y}_i|\boldsymbol{\vartheta}^s)$ is the density of observation i with respect to the parameters sampled at iteration s. The second term, the effective number of parameters, is estimated via

$$\text{efp}_{\text{WAIC}} = \sum_{i=1}^{n} \text{var}_s(\log f(\boldsymbol{y}_i|\boldsymbol{\vartheta})),$$

where we compute a separate variance for each observation i across the S posterior draws.

The LOO measure estimates the predictive density of each individual observation from a cross-validation standpoint; that is, the predictive density when we hold out one observation at a time and use the remaining observations to update the prior. We can use analytical results to estimate this quantity based on the full-data posterior, removing the need to re-estimate the posterior while sequentially holding out each observation. These results lead to the estimate

$$\text{LOO} = -2 \sum_{i=1}^{n} \log \left(\frac{\sum_{s=1}^{S} w_i^s f(\boldsymbol{y}_i | \boldsymbol{\vartheta}^s)}{\sum_{s=1}^{S} w_i^s} \right), \tag{10}$$

where the w_i^s are importance sampling weights based on the relative magnitude of individual i's density function across the S posterior samples. Package **loo** smooths these weights via a generalized Pareto distribution, improving the estimate.

We can also estimate the effective number of parameters under the LOO measure by comparing LOO to the lppd that was used in the WAIC calculation. This gives us

$$\text{efp}_{\text{LOO}} = \text{lppd} + \text{LOO}/2,$$

where these terms come from Equations 9 and 10, respectively. Division of the latter term by two (and addition, vs. subtraction) offsets the multiplication by -2 that occurs in Equation 10. For further detail on all these measures, see Vehtari *et al.* (2016).

Bayes factor

The Bayes factor between two models is defined as the ratio of the models' marginal likelihoods (e.g., Kass and Raftery 1995). Candidate model 1's marginal likelihood can be written as

$$f_1(\boldsymbol{Y}) = \int f(\boldsymbol{\vartheta}_1) f(\boldsymbol{Y} | \boldsymbol{\vartheta}_1) \partial \boldsymbol{\vartheta}_1, \tag{11}$$

where $\boldsymbol{\vartheta}_1$ is a vector containing candidate model 1's free parameters (excluding the latent variables $\boldsymbol{\eta}_i$) and $f()$ is a probability density function. The marginal likelihood of candidate model 2 may be written in the same manner. The Bayes factor is then

$$BF_{12} = \frac{f_1(\boldsymbol{Y})}{f_2(\boldsymbol{Y})},$$

with values greater than 1 favoring Model 1.

Because the integral from Equation 11 is generally difficult to calculate, Lewis and Raftery (1997) describe a Laplace-Metropolis estimator of the Bayes factor (also see Raftery 1993). This estimator relies on the Laplace approximation to integrals that involve a natural exponent. Such integrals can be written as

$$\int \exp(h(\boldsymbol{u})) \partial \boldsymbol{u} \approx (2\pi)^{Q/2} |\boldsymbol{H}^*|^{1/2} \exp(h(\boldsymbol{u}^*)),$$

where $h()$ is a function of the vector $\boldsymbol{u}$, Q is the length of $\boldsymbol{u}$, $\boldsymbol{u}^*$ is the value of $\boldsymbol{u}$ that maximizes h, and $\boldsymbol{H}^*$ is the inverse of the information matrix evaluated at $\boldsymbol{u}^*$. As applied to marginal likelihoods, we take $h(\boldsymbol{\vartheta}) = \log(f(\boldsymbol{\vartheta})f(\boldsymbol{Y}|\boldsymbol{\vartheta}))$ so that our solution to Equation 11 is

$$f_1(\boldsymbol{Y}) \approx (2\pi)^{Q/2} |\boldsymbol{H}^*|^{1/2} f(\boldsymbol{\vartheta}^*) f(\boldsymbol{Y} | \boldsymbol{\vartheta}^*), \tag{12}$$

where $\boldsymbol{\vartheta}^*$ is a vector of posterior means (or other posterior estimate of central tendency), obtained from the MCMC output. In package **blavaan**, we follow Lewis and Raftery (1997) and compute the logarithm of this approximation for numerical stability.

The Bayes factor is sensitive to choice of prior distribution (e.g., Liu and Aitkin 2008; Vanpaemel 2010), so researchers using the Bayes factor are advised to carefully consider their priors. Kass and Raftery (1995) provide popular rules of thumb for interpreting the log-Bayes factor, though the extent to which these rules are meaningful in any specific application is unclear. In general, as the log-Bayes factor increases from 0, we gain increasing support for the first model. The Bayes factor can also be motivated as the extent to which, after observing data, we should revise the prior odds of model 1 being correct versus model 2 being correct; see, e.g., Kass and Raftery (1995) and Rouder, Morey, Verhagen, Province, and Wagenmakers (2016).

Affiliation:

Edgar C. Merkle
Department of Psychological Sciences
University of Missouri
28A McAlester Hall
Columbia, MO, 65211, United States of America
E-mail: merklee@missouri.edu
URL: http://faculty.missouri.edu/~merklee/