Published in final edited form as:
Struct Equ Modeling. 2022 ; 29(3): 452–475. doi:10.1080/10705511.2021.1911657.

Fitting Multilevel Vector Autoregressive Models in Stan, JAGS, and Mplus

Yanling Li, **Julie Wood**, **Linying Ji**, **Sy-Miin Chow**, **Zita Oravecz**
The Pennsylvania State University

Abstract

The influx of intensive longitudinal data creates a pressing need for complex modeling tools that help enrich our understanding of how individuals change over time. Multilevel vector autoregressive (mlVAR) models allow for simultaneous evaluations of reciprocal linkages between dynamic processes and individual differences, and have gained increased recognition in recent years. High-dimensional and other complex variations of mlVAR models, though often computationally intractable in the frequentist framework, can be readily handled using Markov chain Monte Carlo techniques in a Bayesian framework. However, researchers in social science fields may be unfamiliar with ways to capitalize on recent developments in Bayesian software programs. In this paper, we provide step-by-step illustrations and comparisons of options to fit Bayesian mlVAR models using Stan, JAGS and Mplus, supplemented with a Monte Carlo simulation study. An empirical example is used to demonstrate the utility of mlVAR models in studying intra- and inter-individual variations in affective dynamics.

Keywords

Multilevel vector autoregressive models; Bayesian modeling; missing data; affective dynamics

Introduction

Intensive longitudinal data (ILD) — repeated reports from individuals at relatively dense cadences (e.g., minutes, hours, days) — capture the necessary information to quantify fast-evolving intraindividual dynamics. The intensive nature of such data creates a pressing demand for methods that are able to capture nuanced temporal changes and individual differences therein in the presence of missing data. The multilevel vector autoregressive (mlVAR) model, also regarded as a special case of the dynamic structural equation model (DSEM) (see, e.g., Asparouhov, Hamaker, & B.O. Muthén, 2018; Chow, Haltigan, & Messinger, 2010b; Hamaker, Asparouhov, Brose, Schmiedek, & B.O. Muthén, 2018; Schuurman, Ferrer, de Boer-Sonnenschein, & Hamaker, 2016), is one of such methods known for its merits to simultaneously model reciprocal linkages between dynamic processes and individual differences. Its simpler variant, the vector autoregressive (VAR) model, is a key constituent of dynamic network models that have seen increasing uptake

Correspondence concerning this article can be addressed to Yanling Li, The Pennsylvania State University, 420 Biobehavioral Health Building, University Park, PA 16802 or by yxl823@psu.edu.

in areas such as clinical psychopathology (Bringmann et al., 2013), neuroimaging (Gates & Molenaar, 2012), emotion regulation (Bringmann, Ferrer, Hamaker, Borsboom, & Tuerlinckx, 2018; Chow, Haltigan, & Messinger, 2010a), and personality dynamics (Beck & Jackson, 2020; Wright et al., 2019). A quick trend analysis of the combined keywords, "autoregression" and "network" using the Web of Science portal revealed a 10-fold increase in the annual number of relevant publications between 2010 and 2020, in close parallel to the growing prevalence of ILD in recent years.

In stark contrast to areas such as economics, mathematics, engineering, and the physical sciences, multilevel extensions of VAR, namely, mlVAR, are especially pertinent in areas of social and behavioral sciences. In these fields the typically limited number of repeated occasions available from each unit (e.g., individual) renders it imperative to "borrow strengths" from all individuals' data for estimation and inferences. At the same time the substantial between-unit (e.g., between-individual) heterogeneity in dynamics speaks against direct pooling of data without regard to plausible sources of random effects, therefore calling for a multilevel modeling approach. However, fitting mlVAR in a single step is a computationally challenging problem (Epskamp, Waldorp, Mõttus, & Borsboom, 2018), and the broader audience of social and behavioral science remains unfamiliar with some of the recent software advances, specifically in the Bayesian framework (e.g., Asparouhov et al., 2018; Schuurman, Grasman, & Hamaker, 2016), that can facilitate such modeling endeavors. The present article aims to review and compare the pros and cons of three currently available Bayesian software — Stan (e.g., Carpenter et al., 2017), Just Another Gibbs Sampler (JAGS; Plummer et al., 2003) and Mplus (L.K. Muthén & B.O. Muthén, 1998-2017) — for fitting single-step mlVAR models via a Monte Carlo simulation study. Step-by-step illustrations and guidelines on data inputs, model specifications, missing data handling, and model inferences are provided in the context of an empirical example of emotion regulation using data from the Affective Dynamics and Individual Differences (ADID; Emotions and Dynamic Systems Laboratory, 2010) study.

The rest of the paper is organized as follows. We first review rationales for Bayesian mlVAR modeling and previous tutorials on fitting Bayesian mlVAR models. This is followed by introduction of the mlVAR model and the motivating empirical example to help set the context for our illustrations, where we present the mlVAR model considered in the present article and the corresponding Bayesian modeling framework. We then provide step-by-step illustrations of Stan, JAGS and Mplus code using the empirical example, followed by a summary of the empirical results and a simulation study. Finally, we discuss the results and highlight some future directions.

Rationales for Bayesian mlVAR Modeling

Computational hurdles associated with fitting mlVAR and related models under complex, high-dimensional random effect structures are well acknowledged in the classical (frequentist) statistical framework (Chow et al., 2010b; Epskamp et al., 2018; Song & Zhang, 2014). The Bayesian framework offers several advantages when it comes to computational complexity. First, Bayesian parameter estimation utilizes Markov chain Monte Carlo (MCMC) algorithms, which eliminate the need to perform simultaneous high-dimensional integration and greatly simplify the estimation of more complex models (Robert & Casella, 2013; Schuurman, Grasman, & Hamaker, 2016). Second, Bayesian methods are more flexible in dealing with missing data (Daniels & Hogan, 2008; Ji et al., 2020; Tang, Chow, Ibrahim, & Zhu, 2017). Third, Bayesian methods provide accurate quantification of the uncertainty associated with the estimation results by readily propagating multiple sources of uncertainty from the data into the estimation procedure (Oravecz, Tuerlinckx, & Vandekerckhove, 2016). It is also straightforward to obtain summary statistics of estimated model parameters such as means, standard deviations (i.e., standard errors), and credible intervals through Monte Carlo integration (see, e.g., Oravecz & Muth, 2018). Finally, since Bayesian methods do not depend on large sample theory, they have advantages over frequentist methods when the sample size is small (Lee & Song, 2004).

Previous Tutorials on Fitting mlVAR Models

Currently, many Bayesian software programs can be used to fit mlVAR models — some popular ones include Stan, JAGS and Mplus, all of which are available through interfaces in R (e.g., rstan (Stan Development Team, 2020), rjags (Plummer, Stukalov, & Denwood, 2019), and MplusAutomation (Hallquist & Wiley, 2018)). Even though ample tutorial examples and user forums for these software programs exist and can be easily accessed online (e.g., Korner-Nievergelt et al., 2015; Kruschke, 2014; McElreath, 2018; L.K. Muthén & B.O. Muthén, 1998-2017; Sorensen & Vasishth, 2015), few of these resources cover fitting complex mlVAR models for ILD in the presence of missing data. Schuurman (2018) provided JAGS code for fitting bivariate mlVAR models, but did not allow for random innovation covariance matrices or provide ways of handling missing data — both of which are reasonable expectations in empirical scenarios (Almeida, Piazza, & Stawski, 2009; Hamaker et al., 2018; Hedeker & Mermelstein, 2007; Hedeker, Mermelstein, Berbaum, & Campbell, 2009; Wang, Hamaker, & Bergeman, 2012). The former omission is a critical one, as misspecification in innovation covariance structures has been shown to have direct consequences on other parameters in the VAR model (e.g., du Toit and Browne, 2007, p. 81; Canova, 2011, p. 117). In addition, fitting mlVAR models with missing data, even under simpler assumptions that the data are missing at random (MAR) — namely, that the probability of missingness depends only on variables that are observed (Little & Rubin, 1987) — is still non-trivial due to many implicit software-specific restrictions that may be abstruse to researchers interested in, but otherwise unfamiliar with the mlVAR models.

Multilevel Vector Autoregressive (mlVAR) Models

Before introducing the mlVAR model formally, we begin first by introducing one of its simplest and most commonly-known univariate special case, the autoregressive (AR) model — a model describing how a variable in a system predicts itself at the next time point. The AR(p) model is expressed as:

$$y_{i,t} = \phi_0 + \sum_{l=1}^{p} \phi_l \left(y_{i,t-l} - \phi_0\right) + u_{i,t},\ u_{i,t} \sim N\left(0, \sigma^2\right), \tag{1}$$

where i represents the ith person ($i = 1, 2, \ldots, N$), t represents the tth time point ($t = 1, 2, \ldots, T_i$), and p is the order of the AR process. Conceptually, an AR(p) model can be viewed as a linear regression model in which the explanatory variables consist of $y_{i,\,t\,-\,l}(l = 1, \ldots, p)$, namely, the lags of the dependent variable, $y_{i,t}$. The intercept and regression coefficients are denoted by ϕ_0 and $\phi_l(l = 1, \ldots, p)$, respectively. The innovation term (also called process noise), $u_{i,t}$, reflects unmeasured sources that affect the dynamics of $y_{i,t}$, following a normal distribution with zero mean and variance σ^2. An AR(p) process is stationary (Lütkepohl, 2005) when the modulus of all the roots of $1 - \sum_{l\,=\,1}^{p} \phi_l B^l = 0$ are greater than 1, where B is the lag operator such that $By_t = y_{t\,-\,1}$. Model parameters from any AR-type models can be properly estimated if and only if the process is stationary.

In many applications of the AR model, data have been demeaned and often detrended prior to model fitting, thus the intercept term ϕ_0 can be omitted from the above model. In the present study, we choose to include ϕ_0 because its value reflects an individual's baseline around which the process of interest fluctuates and thus can be of explicit interest. When p = 1, the AR(1) model has only one regression coefficient ϕ_1, which has been referred to as the "inertia" of a dynamic process given that a high positive value of ϕ_1 reflects a construct's resistance to change (Brose, Schmiedek, Koval, & Kuppens, 2015; Koval, Kuppens, Allen, & Sheeber, 2012; Kuppens, Allen, & Sheeber, 2010). For instance, when modeling affect dynamics using the AR(1) model, a high positive ϕ_1 indicates that the level of individual's affect is less likely to change or it may take long to recover after a change caused by external events, thus indicating weak affect regulation.

If the relationship between dynamics of multiple outcome variables are of interest, then the VAR model of lag order p (VAR(p) model) can be used to account for the reciprocal linkages among dynamic processes as:

$$\boldsymbol{y}_{i,\,t} = \boldsymbol{\phi}_0 + \boldsymbol{\phi}_1\left(\boldsymbol{y}_{i,\,t\,-\,1} - \boldsymbol{\phi}_0\right) + \ldots + \boldsymbol{\phi}_p\left(\boldsymbol{y}_{i,\,t\,-\,p} - \boldsymbol{\phi}_0\right) + \boldsymbol{u}_{i,\,t},\ \ \boldsymbol{u}_{i,\,t} \sim N(\boldsymbol{0}, \boldsymbol{\Sigma}), \tag{2}$$

where $\boldsymbol{y}_{i,t}$ is a $D \times 1$ (D is the number of outcome variables) matrix of outcome variables; $\boldsymbol{\phi}_0$ is a $D \times 1$ matrix of intercepts; $\boldsymbol{\phi}_l(l = 1, \ldots, p)$ is a $D \times D$ coefficients matrix; $\boldsymbol{u}_{i,t}$ is a $D \times 1$ matrix of innovation terms following a multivariate normal distribution with a fixed covariance matrix $\boldsymbol{\Sigma}$. A VAR(p) process is stationary when all the roots of the determinant of matrix $\mathbf{I} - \boldsymbol{\phi}_1 B - \cdots - \boldsymbol{\phi}_p B^p$ have moduli greater than 1, where B has been defined above.

The specifications above assume that model parameters are the same for each individual, which may not be the case in reality — individuals may have different baseline equilibria as well as patterns of dynamics. One possible way of representing such individual differences is to extend the VAR(p) model in Equation 2 into an mlVAR(p) model in which all level-1 or *within-level* parameters are person-specific, labeled with a subscript i, and come from joint level-2 or *between-level* distributions. The within-level model is specified as:

$$\begin{aligned}\mathbf{y}_{i,\,t} = \boldsymbol{\phi}_{0,\,i} + \boldsymbol{\phi}_{1,\,i}\left(\mathbf{y}_{i,\,t\,-\,1} - \boldsymbol{\phi}_{0,\,i}\right) + \ldots + \boldsymbol{\phi}_{p,\,i}\left(\mathbf{y}_{i,\,t\,-\,p} - \boldsymbol{\phi}_{0,\,i}\right) + \boldsymbol{u}_{i,\,t}, \boldsymbol{u}_{i,\,t} \\ \sim N(\mathbf{0}, \boldsymbol{\Sigma}_i),\end{aligned} \tag{3}$$

where $\boldsymbol{\phi}_{0,i}$ is a $D \times 1$ matrix of *person-specific* intercepts; $\boldsymbol{\phi}_{l,\,i}(l = 1, \ldots, p)$ is a $D \times D$ *person-specific* coefficients matrix; and $\boldsymbol{u}_{i,t}$ is a $D \times 1$ matrix of random innovations following a multivariate normal distribution with a *person-specific* covariance matrix $\mathbf{\Sigma}_i$. The between-level model is expressed as:

$$g(\boldsymbol{\theta}_i) = \mathbf{B}\boldsymbol{x}_i + \boldsymbol{\zeta}_i, \boldsymbol{\zeta}_i \sim N(\mathbf{0}, \mathbf{\Psi}), \tag{4}$$

where $\boldsymbol{\theta}_i$ is an $n_p \times 1$ matrix that consists of all person-specific parameters that appear in $\boldsymbol{\phi}_{0,i} - \boldsymbol{\phi}_{p,i}$ and $\mathbf{\Sigma}_i$ in Equation 3; $g(\cdot)$ is a function used for parameter transformation (see, e.g., the section below for parameter transformations adopted in the present article). Variable $\boldsymbol{x}_i$ is an $n_x \times 1$ matrix of exogenous predictors hypothesized to explain some of the inter-individual differences in $\boldsymbol{\theta}_i$, with the first entry being unity to define an intercept term, and $\boldsymbol{B}$ is the corresponding $n_p \times n_x$ matrix of "between-level" regression coefficients (or fixed effects). Finally, $\boldsymbol{\zeta}_i$ is an $n_p \times 1$ matrix of random effects which represent person i's deviations in the values of $\boldsymbol{\theta}_i$, not accounted for by the exogenous variables; these are distributed with zero means and the random effect covariance matrix $\mathbf{\Psi}$.

Motivating Example: A Bayesian Bivariate mlVAR Model of Affective Dynamics

We used a set of previously published data from the ADID study to investigate intraindividual dynamics of positive affect (PA) and negative affect (NA) — namely, their concurrent and lagged relations, along with inter-individual differences in these relations. We also studied how levels of perceived stress and extraversion may explain some of the individual differences in affect dynamics and baselines. In the ADID study, participants (ages ranging between 18 and 86 years) were asked to rate their momentary emotions five times a day for a little over a month. Self-reported PA and NA were measured via the Positive Affect and Negative Affect Schedule (PANAS; Watson, Clark, & Tellegen, 1988) which consists of two 10-item scales measuring PA and NA respectively. For each item, participants were asked to rate on a 4-point scale the extent to which the affect has been experienced. As in previous studies (Chow & Zhang, 2013; You, Hunter, Chen, & Chow, 2019), we aggregated the ILD to two equally spaced data blocks a day, yielding a total of 26 to 74 measurement occasions per participant. Consistent also with earlier studies, participants with more than 65% of missing data and insufficient variability (i.e., near-zero within-person, over-time standard deviations) in their responses were excluded, yielding a final sample size of 217 participants[1]. To remove the linear trends in PA and NA, we first regressed PA and NA on measurement occasions, respectively, to obtain their residuals, and then added the person-specific means of PA and NA back to the residuals to obtain the final scores for PA and NA to be used in modeling fitting. In addition to momentary emotions, participants' levels of perceived stress (as measured with a 5-item short-form of the Perceived Stress Scale; Cohen, Kamarck, & Mermelstein, 1983) and extraversion (Costa, McCrae, & Dye, 1991), as aggregated from each person's responses across the entire study span and then standardized across persons, were included as predictors of inter-individual differences on the person-specific parameters[2].

[1] Since the data set was used only for illustration purposes, we pre-processed the raw data following the same procedure as implemented in previous studies. Other data pre-processing and exclusion criteria may be used in other studies as appropriate.

To address the questions outlined above, we fitted a bivariate mlVAR(1) model to the data. Corresponding to Equation 3, the *within-level model* was expressed as:

$$\begin{bmatrix} y_{1,i,t} \\ y_{2,i,t} \end{bmatrix} = \begin{bmatrix} \mu_{1,i} \\ \mu_{2,i} \end{bmatrix} + \begin{bmatrix} a_{1,i} & b_{1,i} \\ b_{2,i} & a_{2,i} \end{bmatrix} \left(\begin{bmatrix} y_{1,i,t-1} \\ y_{2,i,t-1} \end{bmatrix} - \begin{bmatrix} \mu_{1,i} \\ \mu_{2,i} \end{bmatrix} \right) + \begin{bmatrix} u_{1,i,t} \\ u_{2,i,t} \end{bmatrix}$$
$$\begin{bmatrix} u_{1,i,t} \\ u_{2,i,t} \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \boldsymbol{\Sigma}_i = \begin{bmatrix} \sigma_{1,i}^2 & r_i \sigma_{1,i} \sigma_{2,i} \\ r_i \sigma_{1,i} \sigma_{2,i} & \sigma_{2,i}^2 \end{bmatrix} \right) \tag{5}$$

where $y_{1,i,t}$ and $y_{2,i,t}$ were PA and NA measurements for person i at time point t. Corresponding to parameters in Equation 3, $\boldsymbol{\phi}_{0,i}$ was specified here as a 2 × 1 matrix of person-specific intercepts with entries $\mu_{1,i}$ and $\mu_{2,i}$; the $\boldsymbol{\phi}_{1,i}$ matrix was specified here as a 2 × 2 coefficient matrix in which $a_{1,i}$ and $a_{2,i}$ represented person-specific auto-regression parameters, and $b_{1,i}$ and $b_{2,i}$ represented person-specific cross-regression parameters; $\boldsymbol{u}_{i,t}$ was a 2 × 1 matrix of random innovations; and $\boldsymbol{\Sigma}_i$ was the random innovation covariance matrix where random innovation standard deviations and correlations were denoted as $\sigma_{1,i}$, $\sigma_{2,i}$ and r_i, respectively.

All person-specific parameters (some with log and Fisher z-transformations, described later) in Equation 5 were collected into a vector $\boldsymbol{\theta}_i = \left[\mu_{1,i}, \mu_{2,i}, a_{1,i}, a_{2,i}, b_{1,i}, b_{2,i}, \log(\sigma_{1,i}), \log(\sigma_{2,i}), \text{Fisher}-\text{z}(r_i)\right]'$ and regressed on levels of perceived stress (x_1) and extraversion (x_2) as:

$$\begin{bmatrix} \mu_{1,i} \\ \mu_{2,i} \\ a_{1,i} \\ a_{2,i} \\ b_{1,i} \\ b_{2,i} \\ \log(\sigma_{1,i}) \\ \log(\sigma_{2,i}) \\ \text{Fisher}-\text{z}(r_i) \end{bmatrix} = \begin{bmatrix} \mu_1 & \alpha_{\mu_1} & \beta_{\mu_1} \\ \mu_2 & \alpha_{\mu_2} & \beta_{\mu_2} \\ a_1 & \alpha_{a_1} & \beta_{a_1} \\ a_2 & \alpha_{a_2} & \beta_{a_2} \\ b_1 & \alpha_{b_1} & \beta_{b_1} \\ b_2 & \alpha_{b_2} & \beta_{b_2} \\ l\sigma_1 & \alpha_{l\sigma_1} & \beta_{l\sigma_1} \\ l\sigma_2 & \alpha_{l\sigma_2} & \beta_{l\sigma_2} \\ z & \alpha_z & \beta_z \end{bmatrix} \begin{bmatrix} 1 \\ x_{1,i} \\ x_{2,i} \end{bmatrix} + \begin{bmatrix} \zeta_{1,i} \\ \zeta_{2,i} \\ \zeta_{3,i} \\ \zeta_{4,i} \\ \zeta_{5,i} \\ \zeta_{6,i} \\ \zeta_{7,i} \\ \zeta_{8,i} \\ \zeta_{9,i} \end{bmatrix}$$
$$\boldsymbol{\zeta}_i \sim N\left(\mathbf{0}, \boldsymbol{\Psi} = \begin{bmatrix} \psi_1^2 & \psi_{12} & \cdots & \psi_{19} \\ \psi_{21} & \psi_2^2 & \cdots & \psi_{29} \\ \vdots & \vdots & \ddots & \vdots \\ \psi_{91} & \psi_{92} & \cdots & \psi_9^2 \end{bmatrix} \right) \tag{6}$$

Corresponding to parameters in Equation 4, $\boldsymbol{\theta}_i$ was a 9 × 1 matrix of person-specific parameters, and $\boldsymbol{B}$ was a 9 × 3 matrix of between-level regression coefficients. Here, we allowed all person-specific parameters to be explained by the two predictors, perceived

[2] Extraversion was measured using three selected questions from the revised NEO personality inventory (NEO-PI-R) on which the participants were asked to rate on a 4-point scale the extent to which they felt *Passive* vs. *Active*; *Unenergetic* vs. *Energetic*; and *Dominant* vs. *Submissive* since the last time they responded to the survey.

stress and extraversion. In practice, some of the coefficients in $\boldsymbol{B}$ may also be fixed at zero to limit the effects of these predictors only to selected person-specific parameters. Finally, $\boldsymbol{\zeta}_i$ in Equation 4 was a 9 × 1 matrix of random effects and $\boldsymbol{\Psi}$ was the random effect covariance matrix.

We imposed transformations on the random innovation standard deviations and correlations in Equation 6 to place them on the real line for regression modeling. These transformations also ensured the positive definiteness of the random innovation covariance matrices, $\boldsymbol{\Sigma}_i$ for all individuals. Specifically, we log transformed the random innovation standard deviations and Fisher z-transformed (Fisher, 1915) the random innovation correlations. The latter was defined as

$$\text{Fisher}-\text{z}(r_i) = \frac{1}{2}\ln\left(\frac{1+r_i}{1-r_i}\right), \tag{7}$$

to yield normally distributed Fisher-z(r_i) in Equation 6[3].

Bayesian Modeling Framework

Suppose that $\boldsymbol{y} = \left\{y_{1,i,t},\ y_{2,i,t},\ i \in [1, N]; t \in [1, T_i]\right\}$ is an array of complete data for two dependent variables for all individuals and time points, which can be partitioned into observed and missing data such that $\boldsymbol{y} = \left(\boldsymbol{y}^{obs}, \boldsymbol{y}^{miss}\right); \boldsymbol{\omega} = \{\boldsymbol{B}, \boldsymbol{\Psi}\}$ is a collection of model parameters where $\boldsymbol{B} = \left\{\mu_1, \alpha_{\mu 1}, \beta_{\mu 1}, \ldots, z, \alpha_z, \beta_z\right\}$ and $\boldsymbol{\Psi}$ are presented in Equation 6.

Then the density of $\boldsymbol{y}$ can be given by

$$f(\boldsymbol{y}|\boldsymbol{\omega}) = \prod_{i=1}^{N}\prod_{t=1}^{T_i} f\left(\boldsymbol{y}_{i,t}|\boldsymbol{y}_{i,t-1}, \boldsymbol{\omega}\right) \tag{8}$$

where $f\left(\boldsymbol{y}_{i,t}|\boldsymbol{y}_{i,t-1}, \boldsymbol{\omega}\right)$ refers to the density of $N\left(\boldsymbol{\phi}_{0,i} + \boldsymbol{\phi}_{1,i}\left(\boldsymbol{y}_{i,t-1} - \boldsymbol{\phi}_{0,i}\right), \boldsymbol{\Sigma}_i\right)$ (see definitions of $\boldsymbol{\phi}_{0,i},\ \boldsymbol{\phi}_{1,i},$ and $\boldsymbol{\Sigma}_i$ in Equation 5).

In the following section, we will discuss how Bayesian methods can be used to handle missing data stemming from three missing mechanisms — missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR) (Little & Rubin, 1987). Specifically, MCAR refers to the scenarios where the missing probability does not depend on any variables, whereas under MAR, the missing probability depends on variables that are observed in the data set. NMAR, also called nonignorable missingness, refers to the scenarios where the missingness depends on information that is unobserved in the data.

Suppose that $\boldsymbol{R} = \left\{R_{1,i,t},\ R_{2,i,t},\ i \in [1, N]; t \in [1, T_i]\right\}$ is an array of missing data indicators, with 0 representing observed entries and 1 representing missing entries; $\boldsymbol{\lambda}$ is a collection of

[3] Although the presented covariance matrix decomposition and the subsequent Fisher-z transformation of the correlation do not generalize to more than 2 dimensions, it was adopted in this specific context to aid computational efficiency, and to provide simple, intuitive interpretations of the effects of predictors on the person-specific correlations (see also Oravecz, Tuerlinckx, & Vandekerckhove, 2011).

parameters related to the density of missing data indicators. Based on the Bayesian selection model (Daniels & Hogan, 2008; Diggle & Kenward, 1994), the distribution of the observed data can be factorized as

$$\begin{aligned} f\left(\boldsymbol{y}^{obs}, \boldsymbol{R} \mid \boldsymbol{\omega}, \lambda\right) &= \int f\left(\boldsymbol{y}^{obs}, \boldsymbol{y}^{miss}, \boldsymbol{R} \mid \boldsymbol{\omega}, \lambda\right) d\boldsymbol{y}^{miss} \\ &= \int f\left(\boldsymbol{y}^{obs}, \boldsymbol{y}^{miss} \mid \boldsymbol{\omega}\right) f\left(\boldsymbol{R} \middle| \boldsymbol{y}^{obs}, \boldsymbol{y}^{miss}, \lambda\right) d\boldsymbol{y}^{miss} \end{aligned} \tag{9}$$

where the first term, $f\left(\boldsymbol{y}^{obs}, \boldsymbol{y}^{miss} \mid \boldsymbol{\omega}\right) = f(\boldsymbol{y} \mid \boldsymbol{\omega})$, is as defined in Equation 8, and it specifies the density of the dependent variables.

The second term defines the density of the missing data indicators in $\boldsymbol{R}$, or in other words, the missing data model. Whenever the missing data mechanism depends on $\boldsymbol{y}^{miss}$, the term $f\left(\boldsymbol{R} \middle| \boldsymbol{y}^{obs}, \boldsymbol{y}^{miss}, \lambda\right)$ cannot be taken out of the integral over $\boldsymbol{y}^{miss}$. As a result, correct inferences on $\boldsymbol{\omega}$ is contingent on correct specification of $f\left(\boldsymbol{R} \middle| \boldsymbol{y}^{obs}, \boldsymbol{y}^{miss}, \lambda\right)$ and inferences with regard to $\boldsymbol{\lambda}$. In this case, the missingness is said to be non-ignorable and has to be handled with caution. In contrast, under MCAR, the missing data mechanism reduces to $f(\boldsymbol{R} \mid \lambda)$ and under MAR, the missing data mechanism can be stated as $f\left(\boldsymbol{R} \middle| \boldsymbol{y}^{obs}, \lambda\right)$. In both of these cases, the term involving $\boldsymbol{R}$ can be taken out of the integral and inferences with regard to $\boldsymbol{\omega}$ are not affected by $\boldsymbol{\lambda}$ or the mising data mechanism. The modeling examples in this article focused on MAR assumptions, but for generality and didactic reasons, we also provided Stan and JAGS code examples that involved one possible way of handling NMAR scenarios via the selection model approach (see the “Step-by-step_Guide_Stan_JAGS.pdf” file in the GitHub repository via https://github.com/yanlingli1/Bayesian-mlVAR).

Prior Specifications.—Prior distributions on the unknown model parameters are integral parts of the Bayesian model because they influence the posterior distributions of parameters, and relatedly, the final estimation results. The between-level model in Equation 6 specified prior distributions on the unknown person-specific parameters. The rest of the unknown parameters were stored in $\boldsymbol{\omega} = \{\boldsymbol{B}, \boldsymbol{\Psi}\}$ as defined above — priors on these parameters are called hyperpriors since they represent priors for parameters in the prior distribution of the person-specific parameters. Specifically, the following weakly informative hyperpriors were specified.

$$\begin{aligned} \text{all parameters in } \boldsymbol{B} &\sim N(0, 100) \\ \boldsymbol{\Psi} &\sim IW(I, 10) \end{aligned} \tag{10}$$

The specification of a relatively small variance (i.e., 100) in the normal distribution was guided by the permissible ranges of auto- and cross-regression coefficients for stationary VAR(1) processes, which often fall within the range of [−1.5, 1.5] (Hamilton, 1994; Lütkepohl, 2005). This specification assigned roughly equal probabilities on the theoretical range of these parameters. The prior of $\boldsymbol{\Psi}$ was specified as an inverse-Wishart (IW) distribution with an identity scale matrix and 10 degrees of freedom (df; i.e., size of $\boldsymbol{\theta}_i +$ 1) so that the correlation parameters had a uniform distribution between −1 and 1 (Alvarez, Niemi, & Simpson, 2014). Note that Stan has a nuanced prior specification option for the covariance matrix, Ψ, which will be discussed in the step-by-step guide below.

Step-by-step Guide to Fitting mlVAR Models in Stan, JAGS, and Mplus

Using data presented above, this section provides a step-by-step guide to fitting the mlVAR model presented above in Stan, JAGS and Mplus. Installation of R is required to implement code examples in the guide. The example data format is displayed below.

ID	Time	y1	y2	x1	x2
1	1	x	x	x	x
1	2	x		x	x
1	3		x	x	x
2	1	x	x	x	x
2	2	x		x	x
…					

where observed data (denoted with x-s) were in long format with columns representing person identifiers, time points, scores on PA, NA, perceived stress and extraversion, respectively. Note that missingness only occurred in PA and NA in the current application, and for each person, the first observation was fully observed (i.e., both PA and NA were observed).

Due to the space limit, we put illustrations of data preparation (i.e., Step 0) for Stan and JAGS, and some relevant R code in our GitHub repository (see the "Step-by-step_Guide_Stan_JAGS.pdf" file). The reproducible code examples based on simulated datasets for all three programs can also be found in the repository.

Stan

After data preparation (i.e., Step 0), dependent variables and between-level predictors will be stored in `Y` and `X`, respectively, where missing entries are replaced with 99 as missing flags. We also create a set of indicator variables (e.g., `TimePoint, Y_miss, Y_miss1, Y_miss2` and `Y_miss3`) in Step 0 to help locate (missing) data entries.

Step 1: Defining the "data", "parameters", and "transformed parameters" blocks.—Stan needs to compile the model code in C (Kernighan & Ritchie, 2006) before the analysis can start. For this, following the convention of C, users have to declare the data types for all variables (e.g., `int` for integers; `real` for values on the `real` line). It is also useful to provide some information on the range of the variables (see, e.g., `<lower=1>`).

```
modelString = "

data {

int < lower =1> P;          // number of persons
int < lower =1> T;          // maximum number of time points

vector [2] Y[P,T];          // dependent variables

int TimePoint [P];          // number of time points for each person

int < lower =1> N_ miss ;    // total number of missing entries

int Y_ miss1 [N_ miss];    // persons who had missing data

int Y_ miss2 [N_ miss];    // time points with missing data

int Y_ miss3 [N_ miss];    // variables with missing data

int < lower =1> K;    // number of covariates (including intercept)

matrix [P,K] X;     // between - level predictors / covariates

vector [2] mus _1;          // mean vector for the 1st obs

matrix [2,2] Sigma _VAR_1;  // covariance matrix for the 1st obs

} // end of the data block
```

All model parameter types (including between-level parameters and person-specific parameters in the within-level model) are declared in the “parameters” block. Just like for the data, we can also declare the ranges of the parameters. For instance, we constrain the range of `sigma` by setting the lower bound to 0, as it represents a vector of standard deviations. According to the missing data modeling strategies illustrated in McElreath (2018), we create a vector `Y_impute` to serve as a place holder for imputed values of the missing entries, the length of which is the number of all missing entries (i.e., `N_miss`). We can see here that missing data are modeled as parameters with probability distributions, which is a useful property of the Bayesian modeling framework. Finally, this part also defines the Cholesky factor of the random effect covariance matrix (see detailed illustration of the Cholesky factor in the Hyperpriors section below).

```
parameters {

vector [N_miss] Y_impute; // a vector of imputed missing values

matrix [P,9] b; // person - specific parameters

matrix [K,9] Coeff; // between-level regression coefficients

vector<lower=0>[9] sigma; // random effect standard deviations

// cholesky factor of the random effect correlation matrix
cholesky_factor_corr[9] L;

} // end of the parameters block
```

The “transformed parameters” block is used to declare parameters that can be derived *deterministically* from the parameters declared in the “parameters” block. To provide an example, while the person-specific parameters `b[pp,9]` (i.e., Fisher-z(r_i) in Equation 6) are declared in the “parameters” block, their transformed values, `corr_noise[pp]` (i.e., r_i), are declared in the “transformed parameters” block.

The last part of this block creates a transformed parameter, `Y_merge`, which stores both observed data and imputed missing data. First, we declare `Y_merge` and pass data `Y` to `Y_merge`; then with index manipulation, we replace all entries valued at 99 in `Y_merge` with the elements in `Y_impute`. As a result, `Y_merge` is composed of observed values in the original data set and parameters representing imputed values of missing data. In the following script, we will focus on the likelihood of `Y_merge` instead of data `Y`.

```
transformed parameters {

// population means of person - specific parameters

matrix [P,9] bmu;

// random innovation covariance matrices

vector < lower = −1, upper = 1 > [P] corr _noise ;

vector [2] sd_noise [P] ;

matrix [2,2] Sigma_VAR[P];

// an array storing observed values in the original data set

// and imputed missing values (i.e., elements in Y_impute)

vector [2] Y_merge [P,T];

for (pp in 1:P) {

// calculate population means of person - specific parameters

bmu [pp] = X [pp,] * Coeff ;

// parameter back - transformations

corr _ noise [pp] = (exp (2 *b [pp,9]) −1) / (exp (2 *b [pp,9]) +1) ;
sd_ noise [pp,1] = exp (b [pp,7]) ;

sd_ noise [pp,2] = exp (b [pp,8]) ;

// make random innovation covariance matrices

Sigma _ VAR [pp,1,1] = sd_ noise [pp,1] *sd_ noise [pp,1];

Sigma_VAR [pp,2,2] = sd_noise [pp,2] *sd_noise [pp,2];

Sigma _ VAR [pp,1,2] = sd_ noise [pp,1] * corr _ noise [pp] * sd_ noise [pp,2] ;

Sigma _ VAR [pp,2,1] = Sigma _ VAR [pp,1,2];

} // close loop over persons

// replace 99 in Y_merge with parameters in Y_impute

Y_merge = Y;

for (u in 1:N_miss){

Y _merge [Y _ miss1[u],Y _ miss2[u],Y _ miss3[u]] = Y _ impute [u];

}

} // end of the transformed parameters block
```

Step 2: Defining the “model” and “generated quantities” blocks.—In this block, we write out the likelihood function for dependent variables as well as specify the priors and hyperpriors for parameters.

Within-level Model.: This part contains the likelihood function — the level-1 distribution on the data point. Within the loop over persons, a multivariate normal distribution is first specified on the first time point for each person. Still in the loop over persons, we then loop over the `TimePoint` indicator to specify the likelihood for the subsequent time points, following the specification in Equation 5. This part also specifies the missing data handling. That is, the missing values in `Y_merge` are imputed based on the likelihood defined below.

```
model {

vector [2] mus [P,T];

for (pp in 1:P){

// the 1st observation
```

```
Y_ merge [pp,1,1:2] ~ multi_normal (mus_1, Sigma_VAR_1) ;

// the likelihood function for dependent variables

for (tt in 2: TimePoint [pp]) {

mus [pp,tt,1] = b [pp,1]+ b [pp,3]*(Y_merge [pp,tt −1,1] −

b [pp,1]) + b [pp,5]*(Y_merge [pp,tt −1,2] − b [pp,2]) ;

mus [pp,tt,2] = b [pp,2]+ b [pp,6]*(Y_merge [pp,tt −1,1] −

b [pp,1]) + b [pp,4]*(Y_ merge [pp,tt −1,2] − b [pp,2]) ;

Y_merge [pp,tt] ~ multi_normal (mus [pp,tt], Sigma_VAR [pp]) ;

} // close loop over time points

// (person loop is not closed here ; see the next part)
```

Between-level Model.: Here we assign population (level-2) distributions on the person-specific parameters, which function as priors on those parameters (see Equation 6). The `multi_normal_cholesky` function defines a multivariate normal density given the mean vector (i.e., `bmu`) calculated in the “transformed parameters” block and the lower-triangular Cholesky factor of the covariance matrix (i.e., `diag_pre_multiply(sigma, L)`).

```
// population distributions on person - specific parameters

b[pp,1:9] ~ multi_normal_cholesky(bmu [pp,1:9],diag_pre_multiply(

   sigma, L));

} // close loop over persons
```

Hyperpriors.: Priors for regression coefficients in the between-level model are specified based on the description in the prior specifications section. Note that in Stan the univariate normal distribution is parametrized via a mean and a standard deviation (not variance). For the random effect covariance matrix, instead of specifying an IW prior on the whole matrix, Stan recommends specifying separate priors for standard deviations and correlations. This practice follows the suggestions from Gelman et al. (2006), where problems with inverse Gamma (IG) and IW priors were discussed and half Cauchy or uniform priors were recommended for random effect standard deviations in hierarchical models. Assuming that the covariance matrix is factorized as $\boldsymbol{\Psi} = \boldsymbol{S}\boldsymbol{L}\boldsymbol{L}^T\boldsymbol{S}^T$ where $\boldsymbol{S}$ is the standard deviation matrix and $\boldsymbol{L}$ is the Cholesky factor of the correlation matrix, the code below shows that an LKJ prior is specified for $\boldsymbol{L}$ and a half-Cauchy prior is specified for the diagonal elements in $\boldsymbol{S}$. Specifically, the shape parameter of the LKJ prior is set to 1 in our example. A shape parameter higher than 1 favors correlations closer to zero, whereas a shape parameter less than 1 favors correlations further away from zero (see Stan User's Guide (Stan Development Team, 2018) for more information). Finally, we do not specify a prior distribution explicitly for `Y_impute` but let Stan use its default prior (i.e., uniform(-Infinity, Infinity)).

```
// priors on between-level regression coefficients

for (kk in 1:K){

for (ww in 1:9){

Coeff [kk,ww] ~ normal (0,10) ;

}}

// priors on elements in the random effect covariance matrix

L ~ lkj_corr_cholesky(1);

sigma ~ cauchy (0, 10);

} // end of the model block
```

The "generated quantities" block is optional and is executed only after the sampling period. In this example, the block is used to obtain bcov (i.e., the random effect covariance matrix) based on `sigma` and `L` so that we can ask Stan to save posterior samples for bcov in the subsequent code.

```
generated quantities {

matrix [9,9] bcorr; // random effect correlation matrix

matrix [9,9] bcov; // random effect covariance matrix

bcorr = multiply_lower_tri_self_transpose (L);

bcov = quad_form_diag (bcorr, sigma);

} // end of the generated quantities block

" # end of "modelString"
```

Step 3: Running the Stan model and extracting the results.—With all blocks in the model script defined, we save `modelString` in the `mlvar.stan` file and then pass this model script, together with data, names of parameters of interest, number of chains (2 in our example), and iterations (25000 in our example) per chain used in the MCMC procedure, to the `stan` function. Stan will use part of these iterations to adjust its sampling algorithms in a "warm-up" phase, and these iterations (5000 in our example) will not be part of the final posterior samples. We choose to generate initial values by Stan with a seed set to 1. Since two chains are specified, we ask for two cores to do parallel computing to reduce the overall computational time. Once the sampling is done, posterior samples for parameters specified in `parameterlist` will be stored in the Stan object, `VARmodel`, and can be extracted with the `mcmc.list` function in the coda package for calculating summary statistics. For this purpose we use our homegrown `zcalc` function, which is available in our GitHub repository (see the “postcalc.R” file). This function can calculate summary statistics including means, medians, modes, standard deviations, 95% credible intervals, effective sample sizes (ESSs), and $\hat{R}$s (see the Empirical Results section for definitions of ESS and $\hat{R}$). Alternatively, the users may also use the `coda` package to calculate summary statistics.

```
writeLines (modelString, con = "mlvar.stan")

parameterlist <- c ("Coeff", "sigma", "bcov",

                   "b", "corr_noise", "Sigma_VAR")

VARmodel <- stan(file = ’mlvar.stan’, data = stan_data, seed = 1,

                 chains = 2, cores = 2, iter = 25000, warmup = 5000,

                 pars = parameterlist)

codaSamples <- mcmc.list(lapply(1:ncol(VARmodel), function (x){

  mcmc(as.array(VARmodel)[, x, ])

}))

resulttable <- zcalc (codaSamples)
```

JAGS

Similar to the data preparation work for Stan, dependent variables and between-level predictors are stored in `Y` and `X`, respectively, except that the missing entries remain in the data set (i.e., no missing flags are needed in JAGS). We also create a set of indicator variables. For instance, `Tseen` stores the location of time points with fully observed cases (i.e., both PA and NA are observed at a particular time point) for each person, while `Tmiss` stores the location of time points with fully missing (i.e., both PA and NA are missing at a particular time point) or partially observed cases (i.e., only PA or NA is observed at a particular time point) for each person. Note that these indicators are not needed when there are no partially observed cases in the data set.

Step 1: Defining models and priors.—The JAGS code for defining models and priors is similar to the Stan code presented above except that JAGS does not require the specification of data types. Given this, we will present the JAGS code and only highlight the differences between Stan and JAGS code.

Within-level Model.: When fitting the model to fully observed data (i.e., looping over `Tseen`), the code follows the same logic as the Stan code. However, when it comes to partially observed cases, it is infeasible to assign a multivariate distribution on `Y[pp,tt,1:2]`. To work around this restriction, for each person we loop over the missing time points (i.e., `Tmiss`) and assign univariate normal distributions on `Y[pp,tt,1]` and `Y[pp,tt,2]` separately[4]. This part also specifies the missing data handling — the missing values are imputed based on the likelihood defined below in the `Tmiss` loop.

```
modelString = "

model{

for (pp in P1) {

# the likelihood function for dependent variables

for (tt in Tmiss[pp,1:nmiss [pp]]){

mus [pp,tt,1] <− b [pp,1] +b [pp,3] * (Y [pp,tt−1,1] −b [pp,1]) +b [pp,5] *
(Y [pp

   ,tt−1,2] −b [pp,2])

mus [pp,tt,2] <− b [pp,2] +b [pp,6] * (Y [pp,tt−1,1] −b [pp,1]) +b [pp,4] *
(Y [pp

   ,tt −1,2] −b [pp, 2])

Y[pp,tt,1] ~ dnorm(mus[pp,tt,1], pow(sd_noise [pp,1], −2))

Y[pp,tt,2] ~ dnorm (mus [pp,tt,2], pow(sd_noise [pp,2], −2))

} # close loop over Tmiss

} # close loop over persons with missing data

for (pp in 1:P) {

# the 1st observation

Y[pp,1,1:2] ~ dmnorm(mus_1, prec_VAR_1)

# the likelihood function for dependent variables
```

[4] Note that this treatment of missing data is limited to two dimensions and do not readily generalize to higher dimensions. Such programming was implemented only to work around the restriction of handling partially observed multivariate nodes in JAGS, and is not recommended in principle.

```
for (tt in Tseen[pp,2:nseen [pp]]){

mus [pp,tt,1] <− b [pp,1] +b [pp,3] * (Y [pp,tt−1,1] −b [pp,1]) +b [pp,5] *
(Y [pp

   ,tt −1,2] −b [pp,2])

mus [pp,tt,2] <− b [pp,2] +b [pp,6] * (Y [pp,tt −1,1] −b [pp,1]) +b [pp,4] *
(Y [pp

   ,tt −1,2] −b [pp,2])

Y[pp,tt,1:2] ~ dmnorm(mus [pp,tt,1:2], prec_VAR[pp,1:2,1:2])

} # close loop over Tseen

# (person loop is not closed here; see the next part)
```

<u>Between-level Model.:</u> Defining the between-level model in JAGS is almost identical to that in Stan, except that Stan uses the standard deviation parameter (or covariance matrix) for the dispersion parameter of the (multivariate) normal distribution while JAGS uses the precision parameter (or precision matrix), which is the inverse of the variance (or covariance matrix). The transformation between them is implemented via the `pow` (or `inverse`) function. In addition, the parameter (back-)transformation code below is also identical to the code in the “transformed parameters” block in Stan.

```
              # population distributions on person-specific parameters


bmu[pp,1:9] <- X[pp,] % *% Coeff

b[pp,1:9] ~ dmnorm(bmu [pp,1:9], bpre [1:9,1:9])


              # parameter back-transformations


corr_noise [pp] <- (exp (2 *b [pp,9]) −1) / (exp (2 *b [pp,9]) +1)

sd_noise [pp,1] <- exp (b [pp,7])

sd_noise [pp,2] <- exp (b [pp,8])

# make random innovation precision matrices

Sigma_VAR [pp,1,1] <- sd_ noise [pp,1] *sd_ noise [pp,1]
Sigma_VAR [pp,2,2] <- sd_noise [pp,2] *sd_noise [pp,2]

Sigma_VAR [pp,1,2] <- sd_noise [pp,1]* corr_noise[pp]*sd_noise[pp,2]

Sigma_VAR [pp,2,1] <- Sigma_VAR [pp,1,2]

prec_VAR [pp,1:2,1:2] <- inverse (Sigma_VAR [pp,1:2,1:2])

} # close loop over persons
```

Hyperpriors.: Defining hyperpriors for between-level regression coefficients in JAGS is almost identical to that in Stan except for the usage of precision parameters as mentioned above. In terms of the random effect covariance matrix, we assign IW priors as described in the prior specifications section.

```
              # priors on between - level regression coefficients

for (kk in 1:K){

for (ww in 1:9) {

Coeff [kk, ww] ~ dnorm (0, .01)

}}

              # priors on the random effect precision matrix

bpre ~ dwish(W, 10)
```

Finally, we define `bcov` and `sigma` below so that we can ask JAGS to save posterior samples for these parameters in the subsequent code.

```
              # obtain the random effect covariance matrix and

              # random effect standard deviations

bcov <- inverse(bpre)

for(ww in 1:9) {
```

```
sigma[ww] <- sqrt(bcov [ww,ww])
}
} # end of the "model" block
" # end of "modelString"
```

Step 2: Running the JAGS model and extracting the results.—To run a JAGS model, the model script, data, and initial values are passed to the `jags.model` function. Here we also set the seed and let JAGS generate the initial values. Similar to Stan, there is an adaptation phase to maximize the efficiency of the sampling algorithm and samples generated during this phase will be discarded. The number of samples in the adaptive phase is specified in `n.adapt`. Then the JAGS object, `jagsModel`, is passed to the `update` function to perform the burn-in phase, length of which is specified in `n.iter`. Samples generated during this phase are also discarded, as they might be influenced by the starting values, this way ensuring that the final set is drawn from a stationary distribution. After the burn-in phase, `jagsModel` is passed to the `coda.samples` function to obtain `n.iter` samples per chain to approximate the posteriors for parameters of interest specified in `parameterlist`. Then we can use our homegrown `zcalc` function or the coda package loaded automatically with `rjags` to obtain summary statistics.

```
writeLines(modelString, con = "mlvar.txt")
initsl <- list(.RNG.name = "base::Wichmann-Hill", .RNG.seed = 1)
inits2 <- list(.RNG.name = "base::Wichmann-Hill", .RNG.seed = 2)
jagsModel <- jags.model(file = "mlvar.txt", data = jags_data,
                        inits = list(inits1,inits2),
                        n.chains = 2, n.adapt = 4000)
update(jagsModel, n.iter = 1000)
parameterlist <- c("Coeff", "sigma", "bcov",
                   "b", "corr_noise", "Sigma_VAR")
codaSamples <- coda.samples(jagsModel,
                            variable.names = parameterlist,
                            n.iter = 20000)
resulttable <- zcalc(codaSamples)
```

Mplus

Step 0: First, install Mplus.—Details are provided on the Mplus website (https://www.statmodel.com). A license number issued by the Mplus team is needed for installation.

Step 1: Defining the DATA, VARIABLE, and ANALYSIS sections.—The data, saved in the "mlVAR.dat" file, are declared at the beginning of the script. Unlike Stan and JAGS, we do not reformat the data but directly use the long format data displayed before, except that the missing entries are replaced by missing flags (i.e., 99 in our example). In the VARIABLE section, the `NAMES` option declares all variables in the same order as in the data set, the `USEVARIABLES` option declares variables used in the analysis, and the `LAGGED` option declares lagged variables with the order of lag in the parenthesis. The `CLUSTER` and `BETWEEN` options are only used in the multilevel modeling framework to declare the variable containing clustering information (i.e., person indicator, ID) and between-level predictors. The `MISSING` option declares which variables contain missing data and the flag of missingness (i.e., 99).

Under ANALYSIS, we declare fitting a two-level model using Bayesian methods. The number of chains is not explicitly declared here because we use the default number, 2. With `PROCESSORS = 2` declared, two cores will be used for parallel computing. To obtain 20000 posterior samples per chain for parameter estimation, we run 40000 iterations per chain because the default number of burn-in iterations in Mplus is half of the total iterations and cannot be changed manually by the users[5]. With `ESTIMATOR = BAYES` specified, Mplus will implement Bayesian analysis and impute missing values based on the model defined in Step 2, which is conceptually similar to how missing values are imputed in Stan and JAGS.

```
DATA: FILE = "mlVAR.dat";

VARIABLE: NAMES = ID Time y1 y2 x1 x2;

          USEVARIABLES = ID y1 y2 x1 x2;

          LAGGED = y1 (1) y2(1);

          CLUSTER = ID;

          BETWEEN = x1 x2;

          MISSING = y1 y2 (99) ;

ANALYSIS: TYPE = TWOLEVEL RANDOM;

          ESTIMATOR = BAYES;
```

[5] With the R package, `MplusAutomation`, users can extract all posterior samples and include part of the first half for estimation, but caution should be taken when deciding the number of burn-in iterations to make sure that the chains are not influenced by the starting values anymore.

```
            PROCESSORS = 2;

            FBITERATIONS = 40000;
```

Step 2: Defining the MODEL and MODEL PRIORS sections.

Within-level Model.: This part defines the within-level model. The first four lines define person-specific auto- and cross-regression parameters, where y1&1 and y2&1 represent lag 1 values of dependent variables. The remaining lines define the random innovation covariance matrices, which is different from what described in Equation 5. The reason is explained below.

At the time of writing, although the existing Mplus code examples allowed for specifying random effects on innovation covariances, the sign of all covariances had to be either positive or negative, as pre-determined by the user (Hamaker, Asparouhov, & B.O. Muthén, 2017)— a strong assumption for many applications, including our illustrative affect example. One way to circumvent this restriction (L.K. Muthén & B.O. Muthén, personal communication, July 12, 2019) in this bivariate scenario is to introduce two latent variables (i.e., `f1` and `f2`), which are latent equivalents of `y1` and `y2`, and allow the random innovation covariance to be modeled by regressing `f1` on `f2`. Following this parameterization, the log residual variance of `f1` and variance of `f2`, labeled respectively as `v1` and `v2` in our code, are linked to the log random innovation variances in the original mlVAR model (see Equation 5) via the parameterizations shown in supplementary material, section A. Our supplementary material also shows how to perform parameter transformations to obtain the random innovation covariances in the original mlVAR model, which can take different signs by definition. Note that when `f1` and `f2` are used as the latent equivalents of `y1` and `y2`, the measurement error variances of `y1` and `y2` are supposed to be 0. However, doing so would lead to slow mixing of the MCMC chains and extremely inefficient estimation (i.e., highly auto-correlated posterior samples). We thus set their values to a small positive constant (i.e., 0.2) in our example.

```
MODEL : %WITHIN%

        ! person - specific auto- and cross-regression parameters

        a1 |y1 ON y1&1;

        a2 |y2 ON y2&1;

        b1 |y1 ON y2&1;

        b2 |y2 ON y1&1;

        ! model random innovation variances and covariances

        f1 by y1@1; y1@0.2;
        f2 by y2@1 ; y2@0.2;

        s | f1 on f2;

        v1 | f1;

        v2 | f2;
```

Between-level Model.: This part defines the between-level model. Here we regress all person-specific parameters defined in the within-level part, except `s, v1` and `v2`, on the predictors and request estimation of intercepts. In addition, variances of and covariances between all person-specific parameters are defined and labeled.

```
        %BETWEEN%

        ! regress person - specific parameters on predictors

        y1 y2 a1 a2 b1 b2 ON x1 x2 (s1-s12);

        ! estimate the population means of person - specific parameters

        [y1 y2 a1 a2 b1 b2] (s13-s18);

        ! random effect variances and covariances

        y1 y2 a1 a2 b1 b2 s v1 v2 (ve1 - ve9) ;

        y1 y2 a1 a2 b1 b2 s v1 v2 with y1 y2 a1 a2 b1 b2 s v1 v2 (c1 -

           c36);
```

Hyperpriors.: The specification of priors is almost identical to JAGS except that IW priors are specified separately for variance and covariance parameters, and Mplus uses the variance parameter for the dispersion parameter of the normal distribution.

```
MODEL PRIORS: s1 - s18 ~ N(0, 100);

              ve1 - ve9 ~ IW(1, 10);

              c1 - c36 ~ IW(0, 10);
```

Step 3: Defining the OUTPUT, SAVEDATA, and PLOT sections.—The `TECH1` option is used to obtain parameter specifications and starting values, and the `TECH8` option requests that the optimization history be printed in the output. The users can define their own output files depending on which information they are interested to know. By specifying `SAVE = FSC0RES(1000 1)`, we request that "plausible values" (Mplus's general term to denote imputed values of latent variables, typically in the form of draws from the posterior distributions of the latent variables; Asparouhov and B.O. Muthén, 2010) be saved, where 1000 is the number of draws and 1 is the number used for thinning. The plausible values are generated after convergence or the user-specified number of MCMC iterations. After that, Mplus continues the MCMC estimation in the first chain only to produce the number of plausible values specified in the SAVEDATA command (L.K. Muthén & B.O. Muthén, personal communication, September 16, 2019). In the PLOT section, we specify `FACTORS = ALL` to request that the plausible values be available for plotting.

```
OUTPUT: TECH1; TECH8;

SAVEDATA: FILE IS result.csv; SAVE = FSCORES(1000 1);

PLOT: TYPE = PLOT3; FACTOR = ALL;
```

Step 4: Running the Mplus model and extracting the results.—Just like before, all summary statistics of the posterior distributions are calculated with our homegrown function, based on the posterior samples extracted via the `MplusAutomation` package. In particular, for parameters related to random innovation variances and correlations, we need to do parameter (back)transformations to obtain the final estimation results. That is, to obtain the regression coefficients associated with the predictors on $\log(\sigma_{1,i}), \log(\sigma_{2,i})$, and Fisher-z($r_i$) in Equation 6, which are not directly estimated as parameters in the specified Mplus model but rather, are functions of the estimated parameters, we need to transform plausible values of `v1, v2`, and `s` to $\log(\sigma_{1,i}), \log(\sigma_{2,i})$, and Fisher-z($r_i$), followed by solving of all relevant regression coefficients using these transformed plausible values as dependent variables in a multivariate regression analysis. More details can be found in supplementary material, section A.

Empirical Results

Following the step-by-step guides presented above, we fitted mlVAR models to our empirical data in Stan, JAGS and Mplus, and used effective sample size (ESS) and $\widehat{R}$ statistics (Gelman, Rubin, et al., 1992) for evaluating the sampling quality. The ESS statistic describes how many posterior draws in the MCMC procedure can be regarded as independent, which is generally lower than the actual posterior sample size. The $\widehat{R}$ statistic describes the ratio of the overall variance of posterior samples across chains to the within-chain variance — values below 1.1 indicate no issues with convergence of separate chains to an underlying distribution. The diagnostic criteria for optimal sampling quality were set as ESS greater than 800 and $\widehat{R}$ below 1.1. Results showed that ESS was greater than 800 for all parameters with Stan and Mplus, and 86% of the parameters with JAGS; the ESSs for the remaining 14% of the parameters were all greater than 600, which can be considered adequate. The $\widehat{R}$ statistic was below 1.1 for all parameters with all programs. In terms of computational time, it took about 40 hours for Stan and about one hour for JAGS and Mplus to finish the model fitting.

Between-level parameter estimation results, including point estimates, standard errors, and 95% credible intervals, obtained from the three software programs are summarized in Table 1. For random effect covariance estimates (i.e., the off-diagonal elements in $\mathbf{\Psi}$), only estimates credibly different from 0 were reported. Generally speaking, the three software programs yielded similar empirical results, except that Stan yielded lower estimates of some random effect standard deviations (see, e.g., $\psi_2 - \psi_8$), compared with JAGS and Mplus. The following empirical results were discussed based on results from JAGS.

The population means of person-specific baselines of PA and NA (i.e., $\mu_{1, i}\ and\ \mu_{2, i}$) were 2.57 and 1.34, respectively[6]. The population means of person-specific standard deviations of PA and NA (i.e., $\sigma_{1, i}\ and\ \sigma_{2, i})$ were 0.93 and 0.91, respectively. In terms of affect dynamics, the population means of person-specific auto-regression parameters (i.e., $a_{1, i}\ and\ a_{2, i})$ were 0.22 and 0.26, and $a_{1,i}$ and $a_{2,i}$ were credibly different from zero for 45% and 54% of the participants, indicating that levels of PA (NA) were positively associated with previous levels of PA (NA). That is, we found moderate inertia in participants' affect dynamics. The population means of person-specific cross-regression parameters (i.e., $b_{1,i}$ and $b_{2,i}$) were −0.05 and −0.03, but for most participants, the corresponding credible intervals were relatively wide (indicating some uncertainty in the estimates) and contained 0. Finally, the population mean of random innovation correlations (i.e., r_i) was −0.25 (calculated via back-transforming the Fisher-z scores, by $r = \frac{e^{2z} - 1}{e^{2z} + 1}$ where z=−0.26), and r_i was credibly different from zero for 52% of the participants, suggesting moderate and negative associations between concurrent PA and NA within individuals. Previous studies yielded inconsistent findings regarding whether PA and NA are independent or not, and the relationship between PA and NA was found to be dependent on affect measures, time scales, as well as contexts (Bradburn, 1969; Diener & Emmons, 1984; Egloff, 1998; Goldstein & Strube, 1994). For instance, PA and NA were found to be independent when measured with PANAS (Egloff, 1998). In contrast, our study found that PA and NA were moderately and inversely correlated within individuals, and as shown below, the magnitude of the standard deviation of random innovation correlations was high and the corresponding 95% credible interval was rather narrow, indicating that there were substantial individual variations in the magnitude of this correlation. To provide more information about the estimated person-specific parameters, we also plotted point estimates and credible intervals of these person-specific parameters for all participants in Figure 1, where the black error bars represented 95% credible intervals that did not contain 0, indicating that the point estimates were credibly different from 0. We can see that most estimated person-specific parameters were credibly different from 0 for more than half of the participants, except for $a_{1,i}$, $b_{1,i}$ and $b_{2,i}$.

Figure 2 shows the distributions of the estimated person-specific parameters across persons, suggesting clear individual differences in many person-specific parameters. For instance,

[6] The estimated population mean of person-specific baselines of PA was calculated based on $\hat{\mu}_1 + \hat{\alpha}_{\mu 1}\bar{x}_1 + \hat{\beta}_{\mu 1}\bar{x}_2$ where $\bar{x}_1\ and\ \bar{x}_2$ were empirical means of $x_{1,i}$ and $x_{2,i}$ across persons. Since $x_{1,i}$ and $x_{2,i}$ have been standardized to zero mean and unit variance, the estimated population mean was equal to $\hat{\mu}_1$ (i.e., the estimate of μ_1 in Table 1). The estimated population means of other person-specific parameters were calculated in a similar way.

participants were substantially different in their baselines of PA and NA, inertia in PA and NA, as well as correlations between concurrent PA and NA. The random effect standard deviation estimates in the last part of Table 1 (i.e., $\psi_1 - \psi_9$) also showed that there was a remarkable amount of variability in these person-specific parameters. In terms of between-level predictors, person-level characteristics of perceived stress and extraversion were found to be credibly linked to variability in some of those person-specific parameters. Specifically, consonant with previous research linking high inertia in NA to perceived stress and depression (Brose et al., 2015; Koval et al., 2012; Kuppens et al., 2010; Watson, 1988), we found that participants who reported higher levels of perceived stress had higher inertia in NA (α_{a2} = 0.04, 95% CI = [0.01, 0.08]), as well as higher baselines of NA ($\alpha_{\mu2}$ = 0.27, 95% CI = [0.23, 0.31]). In addition, consistent with previous findings (Lucas, Le, & Dyrenforth, 2008; Watson & Clark, 1997), participants who were higher in extraversion showed higher inertia in PA (β_{a1} = 0.04, 95% CI = [0.002, 0.07]) and higher baselines of PA ($\beta_{\mu1}$ = 0.21, 95% CI = [0.15, 0.27]). Participants' levels of perceived stress (extraversion) were not credibly linked to the variability in baselines of PA (NA) and inertia in PA (NA); participants' levels of perceived stress and extraversion were not credibly linked to the variability in the variances of PA and NA, correlations and cross-lagged relationships between PA and NA.

Finally, the random effect covariance estimates showed associations between some person-specific parameters, although the credible intervals of some random effect covariances had lower/upper bounds close to 0. For instance, there were positive associations between individuals' baseline levels of PA and NA (see, ψ_{12}), inertias in PA and NA(ψ_{34}), and cross-lagged relationships between PA and NA (ψ_{56}), indicating that participants who had higher baseline PA and higher inertia in PA would also have higher baseline NA and higher inertia in NA. In contrast, negative associations were found between the inertias of the two emotions and their respective innovation standard deviations (see, ψ_{37} and ψ_{48}). As the inertia and innovation standard deviation parameters are both supposed to contribute positively to increasing the system's total variance (du Toit & Browne, 2007; Harvey, 1990), these negative associations might suggest that the person-specific innovations helped capture sources of inter-individual differences in total variance that were not adequately accounted for by the person-specific lag-1 auto- and cross-regression coefficients.

Simulation Study

Simulation Designs

The goal of the simulation study was to compare the performance (e.g., estimation accuracy and computational efficiency) of the three software programs in fitting mlVAR models with missing data under the MAR condition. We considered two sample size configurations, namely, $N = 100$, $T = 60$ and $N = 60$, $T = 100$. The first condition was chosen to mirror the median time series length in the empirical study, while the second configuration was selected as a comparison to show how the sample size and time series length might influence estimation accuracy, sampling efficiency, and convergence in the MCMC procedure. We also considered two conditions in terms of the magnitude of autoregression. The condition with moderate autoregressive values was called the "low-stability" condition, while the condition with relatively higher autoregressive values was called the “high-stability” condition. This design yielded 2×2 conditions in our simulation study, and for each condition, we ran 100 Monte Carlo replications[7].

When simulating data, the complete data were first simulated mostly based on the model presented in Equations 5 and 6, except that only one covariate was included in the between-level model, which was simulated from a standard normal distribution. The true values of model parameters were set based on estimation results from fitting AR models in previous studies (e.g., Chow et al., 2010b; Chow & Zhang, 2013; You et al., 2019). Specifically, the true values under the low-stability condition are summarized below. For the high-stability condition, we changed a_1 and a_2 (highlighted in the matrix) to 0.6 and 0.5 to increase the autocorrelations in the data. For both conditions, variances of person-specific auto- and cross-regression parameters were set to relatively small values (i.e., 0.01) to ensure stationary processes.

$$\begin{bmatrix} \mu_1 & \alpha_{\mu_1} \\ \mu_2 & \alpha_{\mu_2} \\ \boldsymbol{a_1} & \alpha_{a_1} \\ \boldsymbol{a_2} & \alpha_{a_2} \\ b_1 & \alpha_{b_1} \\ b_2 & \alpha_{b_2} \\ l\sigma_1 & \alpha_{l\sigma_1} \\ l\sigma_2 & \alpha_{l\sigma_2} \\ z & \alpha_2 \end{bmatrix} = \begin{bmatrix} 0.5 & -0.3 \\ 0.5 & 0.3 \\ \mathbf{0.3} & -0.2 \\ \mathbf{0.3} & 0.2 \\ -0.1 & -0.2 \\ -0.1 & 0.2 \\ 0.5 & -0.3 \\ 0.5 & 0.3 \\ 0.5 & -0.3 \end{bmatrix}, \boldsymbol{\Psi} = \begin{bmatrix} 0.09 & & & & & & & & \\ -0.045 & 0.09 & & & & & & & \\ 0.015 & 0 & 0.01 & & & & & & \\ 0 & 0.015 & 0 & 0.01 & & & & & \\ 0 & 0 & 0 & 0 & 0.01 & & & & \\ 0 & 0 & 0 & 0 & 0 & 0.01 & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.09 & & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.09 & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.09 \end{bmatrix}$$

Then missingness in two dependent variables were generated based on the missing data mechanism specified below.

$$\begin{aligned} P\left(R_{1,i,t} = 1 \middle| x_{1,i,t}, x_{2,i,t}\right) = logit^{-1}\left(-1.1 + 0.6x_{1,i,t} + 0.6x_{2,i,t}\right) \\ P\left(R_{2,i,t} = 1 \middle| x_{1,i,t}, x_{2,i,t}\right) = logit^{-1}\left(-1.1 + 0.6x_{1,i,t} + 0.6x_{2,i,t}\right) \end{aligned} \tag{11}$$

where the probability of missingness was dependent on two fully observed variables, $x_{1,i,t}$ and $x_{2,i,t}$, simulated from a uniform distribution, $U[-3, 3]$, thus yielding an MAR condition. The data generation code, including missing data generation code, and other reproducible materials are available in our GitHub repository (see materials in “MAR” and “NMAR” folders).

When fitting the model, the model scripts were almost identical to those illustrated in the step-by-step guides, except that (1) only one covariate was included in the between-level

[7] Ideally, more Monte Carlo replications need to be run to generate more reliable results. In this study, we chose 100 replications due to the high computational time. To verify that 100 replications were sufficient for the simulation purpose, we ran 500 replications with JAGS under the N=100, T = 60, and high-stability condition, and the Monte Carlo standard error (MCSE; see Equation 15) did not change substantially compared with that with 100 replications. We note that some previous studies on Bayesian VAR models also used 100 replications (see, e.g., Huber & Feldkircher, 2019; Ji et al., 2020).

model; (2) the mean vector of the distribution for the first observation was set as `mus_1 = c(0,0)`; (3) narrower priors were assigned to unknown parameters based on the more restricted ranges of the simulated data. Specifically, priors for all between-level regression coefficients were assigned to be $N(0, 1)$. In terms of the random effect covariance matrix, $\mathbf{\Psi}$, with Stan, we specified `L~lkj_corr_cholesky(2)` and `sigma~cauchy(0, 2)`; with Mplus, we assigned $IW(W, 12)$ with a diagonal scale matrix W = diag(0.5, 0.5, 0.1, 0.1, 0.1, 0.1, 0.5, 0.5, 0.5) to $\mathbf{\Psi}$; with JAGS, we only freely estimated a subset of covariances (i.e., $\psi_{12}, \psi_{13}, \psi_{14}, \psi_{23}, \psi_{24}, \psi_{34}$) while fixing the remaining covariances in $\mathbf{\Psi}$ to 0 because we found that when freely estimating all elements in $\mathbf{\Psi}$, JAGS would be very inefficient, indicated by rather low ESSs for most parameters. Therefore, given the true structure of $\mathbf{\Psi}$ defined above, we assigned $IW(W, 7)$ with a diagonal scale matrix W = diag(0.5, 0.5, 0.1, 0.1) to the first 4-dimensional submatrix of $\mathbf{\Psi}$, and $U(0, 1)$ to the remaining standard deviations in $\mathbf{\Psi}$. We note that the priors could be made more diffuse if researchers do not have plenty of prior knowledge about the range of possible values that a certain parameter could take.

We calculated summary statistics such as biases, relative biases, standard errors (SEs), Monte Carlo standard errors (MCSEs), root-mean-square errors (RMSEs), sensitivity, and coverage rates for each parameter. We also calculated the average ESS across replications, the percentage of replications with ESS greater than 800 (%ESS > 800), and the percentage of replications with $\hat{R}$ less than 1.1 (%Rhat < 1.1), to evaluate the performance of the samplers and conduct convergence diagnostics. Note that for person-specific parameters, the summary statistics were first calculated for each person and then averaged across persons.

Suppose that M replications were implemented in the simulation study. In the mth replication, the point and standard error estimates of θ were $\hat{\theta}_m$ and $SE_{\hat{\theta}_m}$ respectively. Let the average of point estimates across replications be $\bar{\theta}$, then the bias, relative bias, SE, MCSE and RMSE were defined as follows:

$$\text{bias} = \frac{1}{M}\sum_{m=1}^{M}\left(\hat{\theta}_m - \theta\right), \tag{12}$$

$$\text{relative bias} = \frac{1}{M}\sum_{m=1}^{M}\frac{\hat{\theta}_m - \theta}{\theta}, \tag{13}$$

$$\text{SE} = \frac{1}{M}\sum_{m=1}^{M}SE_{\hat{\theta}_m}, \tag{14}$$

$$\text{MCSE} = \sqrt{\frac{1}{M-1}\sum_{m=1}^{M}\left(\hat{\theta}_m - \bar{\theta}\right)^2}, \tag{15}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left(\hat{\theta}_m - \theta\right)^2}. \tag{16}$$

In terms of other summary statistics, definitions of ESS and $\hat{R}$ have been provided in the previous section. Sensitivity was defined as the proportion of replications in which the credible intervals did not contain 0. The coverage rate was defined as the percentage of replications in which the credible intervals contained the true values.

Simulation Results

Full simulation results can be found in Tables S1 – S15 in supplementary material, section B, including summary statistics for between-level and person-specific parameters for all programs under all conditions. In the main text, we focused on highlighting aspects that suggested notable differences in performance between the software programs. Figure 3 shows comparisons of four selected summary statistics (i.e., relative biases, RMSEs, coverage rates, and %ESS > 800) for between-level regression coefficients across programs under low- and high-stability conditions, with a sample size of N= 100 and T= 60; Figure 4 shows these comparisons with a sample size of N= 60 and T= 100. Due to the complex structure of mlVAR and thus the large number of parameters to compare, we divided all between-level regression coefficients into five categories — intercept-related parameters, auto-regression-related parameters, cross-regression-related parameters, (random innovation) standard deviation-related parameters and (random innovation) correlation-related parameters, and then calculated the average of summary statistics for parameters within the same category. For instance, intercept-related parameters include μ_1, α_{μ_1}, μ_2, and α_{μ_2}. In the following discussion about the performance on each summary statistic, we will start with consistent findings across programs and conditions, and then discuss comparisons between programs, sample size configurations as well as low- and high-stability conditions.

Relative bias: Generally, all software programs yielded satisfactory performance in terms of relative biases (i.e., between −.1 and .1) for most parameters under all conditions. Increasing the sample size helped reduce the relative biases for most parameters with all programs (see the simulation results with N= 200 and T= 100 (i.e., Tables S5, S10 and S15) in supplementary material, section B). Across all parameters, Stan consistently yielded similar or lower relative biases compared to JAGS and Mplus. JAGS had the worst performance on relative biases for cross-regression-related parameters, which might be due to the workaround to handle partially observed multivariate nodes in JAGS. We have verified post-hoc that when there were no partially observed cases in the data and thus a multivariate normal distribution could be assigned to all nodes (either fully observed or fully missing) in JAGS, the biases were reduced substantially. Mplus had the worst performance on standard deviation- and correlation-related parameters, which might be related to the parameter transformation procedure related to these parameters, which yielded a two-step estimation. It might also be related to the small fixed measurement error variance in the model linking latent variables (i.e., `f1` and `f2`) to observed variables (`y1` and `y2`), which should be person-specific in reality but was fixed to a constant for the simplicity of model specifications. We have verified post-hoc that those biases could be reduced with larger sample sizes (see Table S15 in supplementary material, section B). In addition, the comparison between two different sample size configurations showed that Mplus yielded smaller relative biases for standard deviation- and correlation parameters under $N = 60$ and $T = 100$ than under $N = 100$ and $T = 60$. In terms of the comparison between low- and high-stability conditions, higher relative biases for intercept-, auto-regression-, and cross-regression-related parameters were observed under the high-stability condition in all three programs, which was expected because there were more instances of time series that were close to the non-stationary range under the high-stability condition.

RMSE: All software programs yielded satisfactory performance in terms of RMSEs (i.e., < .1) for all parameters under all conditions. Intercept-related parameters had larger RMSEs than other parameters mainly because of larger SE estimates. Similar to relative biases, JAGS yielded slightly larger RMSEs on cross-regression-related parameters and Mplus yielded larger RMSEs on standard deviation- and correlation-related parameters, which, as discussed before, can be improved with larger sample sizes. In general, the comparison between sample size configurations showed slightly higher RMSEs under $N = 60$ and $T = 100$ than under $N = 100$ and $T = 60$ for most parameters with most programs. The RMSEs under the high-stability condition were similar to or higher than those under the low-stability condition.

Coverage rate: Overall, coverage rates for most parameters were close to the nominal rate of .95 (the red dashed lines in Figures 3 and 4) with all programs under all conditions. Basically, the performance on coverage rates were consistent with relative biases because parameters with higher relative biases were expected to have lower coverage rates. In a similar vein, the coverage rates could be improved with larger sample sizes. Note that intercept-related parameters had good coverage rates even though they were a bit biased under the high-stability condition, which might be due to the larger SE estimates and consequently broader credible intervals.

***%ESS* > 800:** In general, ESSs were found to be higher under $N = 60$ and $T = 100$ than under $N = 100$ and $T = 60$ with all programs. Stan yielded the highest ESSs for all parameters under all conditions, compared with JAGS and Mplus. For JAGS, low ESSs were observed for intercept-related parameters under all conditions, and ESSs were rather low under the high-stability condition. Based on our experience, it might be due to the low efficiency of algorithms adopted in JAGS in handling high-dimensional random effects[8]. For Mplus, ESSs were found to be higher for intercept-related parameters but lower for auto- and cross-regression-related parameters under the low-stability condition than under

[8] By checking the samplers used in JAGS, we found that when all person-specific parameters were specified to be correlated, JAGS would use an adaptive random walk Metropolis algorithm called `bugs::MNormal` for these correlated parameters. According to the JAGS manual, this algorithm can be very inefficient and may require an extremely long adaptation period, which has been verified by our simulation, where about 20% of the replications with JAGS did not finish the adaptation phase within 4000 iterations. In addition, we found that when only person-specific intercepts were specified to be correlated to yield a 2-dimensional covariance matrix, JAGS would use a Gibbs sampling algorithm called `bugs::ConjugateMNormal`, which was more efficient, as indicated by much higher ESSs.

the high-stability condition. There were multiple factors that might have affected Mplus's performance on ESS. First, in the model linking latent variables (i.e., `f1` and `f2`) to observed variables (`y1` and `y2`), although we purposefully set the error variance to 0.2 to avoid slow mixing of the MCMC chains, they were still relatively small. Second, under the current small sample size configuration, we purposefully chose a more informative IW prior for the random effect covariance matrix, which successfully reduced biases related to random effect variances and covariances but led to lower ESSs. We verified post-hoc that increasing the magnitude of the error variance and specifying more diffuse priors helped improve ESSs substantially. Therefore, caution should be exercised when comparing the computational efficiency across programs.

In addition to the between-level regression coefficients, Figure 5 shows comparisons of RMSEs for random effect standard deviations and covariances across programs and conditions. Generally, all programs yielded satisfactory performance in terms of RMSEs (i.e., < .1) for all parameters under all conditions. Compared with JAGS and Mplus, Stan yielded slightly higher RMSEs for standard deviations of random intercepts under the high-stability condition. By checking relative biases for these parameters, we found that Stan tended to underestimate these parameters under the high-stability condition (see Tables S2 and S4 in supplementary material, section B). In addition, compared with other parameters, all programs yielded higher relative biases in terms of non-zero random effect covariances (i.e., φ_{12}, ψ_{13}, and ψ_{24}), but such bias could be reduced with larger sampler sizes (see Tables S5, S10, and S15 in supplementary material, section B).

Comparisons of other summary statistics such as biases, SEs, MCSEs, sensitivity and $\hat{R}$ were not presented here, either because results were relatively similar across software programs or because the comparisons of these summary measures were similar to the summary statistics discussed above. For instance, Stan, JAGS and Mplus had similar performance on SEs, MCSEs, and sensitivity, and as expected, smaller SEs and higher sensitivity were obtained with larger sample sizes; conclusions concerning comparisons of biases were similar to those associated with relative biases; $\hat{R}$ was below 1.1 for all parameters in all replications with Mplus. For Stan and JAGS, although not all replications yielded $\hat{R}$ below 1.1, there were only 1 to 2 replications with $\hat{R}$ greater than 1.1 for some parameters.

Finally, it should be noted that the comparisons across programs in this simulation study were based on relatively small sample size configurations due to the high computational cost. With larger sample sizes, it is expected to see reduction in biases and improvement of ESS, as mentioned before.

Comparisons across Software Programs

Generally, all three software programs yielded satisfactory performance across all the conditions considered in our simulation study and we have seen improvement of performance with larger sample sizes. Nevertheless, there are some nuanced differences in the strengths and limitations of these programs, as summarized in Table 2 and elaborated below.

Accessibility

In terms of access, Stan and JAGS are open-source programs available for free, whereas Mplus is a proprietary software. The relative benefits and costs of open-source vs. proprietary software are much-debated in the field, with issues including, but not limited to, accessibility due to cost, availability of technical support, product stability, and transparency of what is happening “under the hood” during data analysis.

R Packages

We used the `rstan` package as an R interface to Stan in our example. For JAGS, in addition to the `rjags` package used in our example, one can opt for `R2jags` (Su & Yajima, 2020) or `runjags` (Denwood et al., 2016), which have largely overlapping functionalities. For Mplus, model fitting and extraction of outputs can be automated via the `MplusAutomation` package. There is also a file, mplus.R (available via https://www.statmodel.com/mplus-R), that provides a set of R functions for extracting plot data and viewing plots by reading the GH5 file created by Mplus.

Model Scripts

The coding requirements for Stan and JAGS are more intense. In particular, Stan users need to declare data types for variables before using them. Mplus has the most concise code, and will be most familiar to users who already use its other frequentist-related functionalities.

Model Specifications

Stan and JAGS have greater flexibility in terms of model specifications than Mplus. In principle, Stan and JAGS users can fit arbitrarily complex models as long as they can specify the likelihood function. For instance, as discussed before, there needs to be some work around to allow for polarity of the random innovation correlations and then regress them on predictors in Mplus. However, such model specification can be easily done in Stan and JAGS (e.g., `bmu[pp] = X[pp,]*Coeff`). Additionally, Stan and JAGS allow users to use `for` loops to automate generalization of VAR models to multiple dimensions and higher orders, as needed. The `for` loops are not supported in Mplus, but Mplus has its own unique coding shortcuts that make generalization to higher dimensions and orders possible with relatively concise coding.

Prior Specifications

Mplus uses conjugate priors (i.e., the conditional posterior distributions and prior distributions are in the same family) as default priors to allow for fast computation. Using conjugate priors has some known advantages (e.g., familiarity and tractability of the forms of the posterior distributions), but also some known drawbacks in the context of multilevel modeling (Gelman et al., 2006). In addition to default priors, Mplus also provides some alternative options for priors, as listed in the user’s guide (L.K. Muthén & B.O. Muthén, 1998-2017). Stan and JAGS provide more flexibility as users can choose prior types from a large pre-defined set of probability distributions. For instance, in Stan there are specialized priors for standard deviation and correlation parameters of a covariance matrix. Additionally, Stan and JAGS users can write their own probability distribution functions (see, e.g., Stan

Development Team, 2018; Wabersich & Vandekerckhove, 2014). While Stan and Mplus have default priors with diffuse/non-informative specifications, informative priors can be set in all three programs by changing the parameter values for priors.

Missing Data Inputs

Stan does not allow any missing entries in the data set. Model fitting with missing data is still possible, but requires replacement of all missing entries by missing flags (e.g., 99 in our example) before passing data to Stan, and then replacing these missing flags by a vector parameter (e.g., `Y_impute` in the example) which stores imputed missing values in the model fitting procedure. With missing variables and missing flags declared, Mplus can automatically handle missing entries. Missing flags are not needed in JAGS, but JAGS does not allow specifications of multivariate distributions on partially observed nodes. Given this, partially observed cases must either be deleted or modeled separately.

Missing Data Handling Techniques

The missing data handling approach adopted in the illustrative example is analogous to a Bayesian full-information likelihood approach for handling missingness, which is known to work adequately under MAR or MCAR. Other missing data handling options, including ways of handling non-ignorable missingness (see, e.g., Daniels & Hogan, 2008; Ji et al., 2020; Lee & Tang, 2006; Li et al., 2019; Tang et al., 2017; van Buuren, Brand, Groothius-Oudshoorn, & Rubin, 2006) is possible in all three programs. Mplus generally provides more restricted missing data modeling options to make coding much easier for the users. For instance, not all of the missing data models currently available in Mplus (e.g., pattern mixture models; B.O. Muthén, Asparouhov, Hunter, & Leuchter, 2011) are supported for use with DSEMs in the latest version of Mplus (i.e., version 8.6), but this may change in future versions. With Stan and JAGS, users need to write their own code to specify the hypothesized missing data mechanisms, models, and distributional assumptions. For instance, in Stan and JAGS, alternative missing data models for covariates may be specified, some examples of which include sequentially and fully conditional models for specifying dependencies among covariates (Ji et al., 2020; Lipsitz & Ibrahim, 1996; Raghunathan, Lepkowski, van Hoewyk, & Solenberger, 2001; van Buuren, 2007).

In addition, in many circumstances, users may be interested in examining posterior values of the imputed missing data values — for instance, to make inferences about longitudinal trends in observed or latent variable scores. In Stan and JAGS, users can monitor and obtain these posterior values in the model fitting procedure. In Mplus, users may obtain imputed values of the missing data through the `DATA IMPUTATION` command, but unlike Stan and JAGS, these are samples generated from one of the chains after convergence or the user-specified number of MCMC iterations.

MCMC Algorithms and Computational Efficiency

Generally speaking, all three programs use variants of the Metropolis-Hastings (MH) algorithm (Hastings, 1970; Metropolis, Rosenbluth, Rosenbluth, Teller, & Teller, 1953) for estimation, which iteratively draw samples from a proposal function and then accept or reject the new values based on the acceptance probability function. Despite this general parallel, the differences across these programs are substantial. To start with, Stan uses Hamiltonian Monte Carlo (HMC; Neal, 2011), a variant of MH that is not available in JAGS or Mplus. It relies on an efficient Markov transition as a proposal function to implement more efficient exploration of the target density, the goal of which is to yield better computational efficiency (especially for high dimensional problems) and more reliable estimation results (Betancourt, 2017) than traditional MH algorithms. Specifically, the distances between successive points generated in HMC are typically larger than those in traditional MH algorithms, thus fewer iterations are needed to obtain sufficient ESSs. Our simulation study showed that with the same number of MCMC iterations, Stan generally yielded the best overall performance across multiple performance measures compared with JAGS and Mplus. Second, JAGS uses Gibbs sampling in general, and when the posterior distributions do not have closed form solutions, it will implement other sampling algorithms such as slice sampling (Neal, 2003), which adaptively chooses the step size to match the local properties of the target density function and thus is more efficient than simple Metropolis updates. This approach may yield serial statistical dependence among samples and therefore may need more iterations to converge.

In terms of actual computational time, based on the same simulated data set consisting of values for 100 persons and 60 time points, we ran the algorithms with two chains, 25000 iterations each. On an Intel i5-8350U, 16GB RAM, Windows 10 computer, Mplus performed the fastest (about 20 minutes), followed by JAGS (about one hour) and Stan (about 6 hours). Stan is computationally more expensive because while the HMC algorithm is more efficient (i.e., requiring fewer “steps” to stabilize on a parameter estimate), each “step” is more time-consuming compared to other MCMC algorithms. Moreover, the overall computational times reported above for Stan and Mplus were based on parallelization, which can help reduce the overall time. In contrast, there is no direct option to do parallel computing in `rjags`.

Diagnostics

Stan provides the most detailed diagnostic “warning” messages about problems or suboptimal behaviors in the MCMC procedure, which may result in biased inferences (Betancourt, 2016; Monnahan, Thorson, & Branch, 2017). For instance, Stan warns if the ESS for the bulk of the distribution and/or for the tails is too low, or if the model does not converge; more details on the warnings are provided here https://mc-stan.org/misc/warnings.html. Mplus provides warning messages when the model does not converge, as well as trace and autocorrelation plots of posterior samples if `TYPE = PLOT2` is declared in the `PLOT` section. JAGS only provides warning messages if the adaptive phase was not long enough for the sampler to optimize its behavior. Therefore, JAGS users are recommended to check the convergence and ESS by calling additional functions (e.g., the `zcalc` function used in our example or functions in the coda package) to obtain posterior distributions.

Discussion

The mlVAR model offers a helpful tool for quantifying and elucidating within-individual dynamics and corresponding individual differences, but is often computationally intractable in the frequentist framework. In this paper, we provided a step-by-step guide to fitting a bivariate mlVAR model in a Bayesian framework in three different software programs - Stan, JAGS and Mplus. In an empirical example, we modeled PA and NA dynamics with the bivariate mlVAR model and found some new theoretical insights in terms of how perceived stress and extraversion contributed to individual differences in affective dynamics. Our simulation results suggested generally satisfactory performance of all three software even under the the small sample size considered (i.e., $N = 100$ and $T = 60$; $N = 60$ and $T = 100$), with only slight decrements in performance in the high- than the low-stability condition, presumably due to the greater estimation challenges presented by the former in having more replications exhibiting dynamics close to the non-stationary range. Of the three programs, Stan displayed the best overall performance in terms of estimation accuracy, coverage rates, and sampling efficiency (as indexed by ESS); JAGS showed slightly worse performance in some of the cross-regression-related parameters, but as mentioned before, when the dataset did not contain partially observed cases, JAGS could yield good performance; Mplus showed greater biases in the estimation of random innovation standard deviation- and correlation-related parameters. As elaborated earlier, Mplus's slightly worse performance in this regard might be related to the two-step procedure we had to invoke in Mplus to circumvent its polarity constraints on the random innovation covariances. Future versions of Mplus may be able to remove or ease up this restriction to improve these use cases.

There are several limitations of the current work. First, we implemented Fisher-z transformation on the correlation parameter of the two-by-two random innovation covariance matrices. When no covariates are used to predict those correlation parameters, we can directly assign priors (e.g., uniform or LKJ priors) to them (Gelman et al., 2006; Lewandowski, Kurowicka, & Joe, 2009). However, when predictors are present, the means of the priors would consist of products of the regression coefficients and covariates, thus making the process of prior assignment less intuitive. Given this, the Fisher-z transformation was adopted as a simple alternative to aid interpretability. We also note that decomposing a person-specific covariance matrix into standard deviations and a correlation, and then assigning priors to those, works well in this bivariate scenario, although such an approach does not generalize to higher dimensions. Second, parameters in the distribution of the first observation were fixed, so there might be some slight misspecification in the initial conditions. Alternatively, users can freely estimate them or specify model implied means and covariance matrix and then estimate them (see, e.g., du Toit & Browne, 2007). Third, only mlVAR models of lag order one were considered in the empirical study. Higher-order autocorrelations may exist in affect dynamics in our empirical illustration.

Although the tutorial example was limited to the bivariate version of mlVAR, generalization to higher-dimensional mlVAR models can be done in a relatively straightforward manner when no predictors are needed to predict random innovation variances and correlations. For instance, in Stan, the random innovation covariance matrix Σ_i can be parameterized similarly to the parameterization of the high-dimensional random effect covariance matrix Ψ in the code example. After that, to modify the script to fit a D-dimensional mlVAR model, the major work would involve replacing 2 to D when looping over `dd` and `dd2` dimensions. If the data set does not contain partially observed cases, the generalization to higher-dimensional mlVAR models can be done in JAGS in a similar way. In addition, the tutorial example can be easily customized for fitting less complex models such as mlVAR with fixed cross-regression parameters or a fixed innovation covariance matrix in Stan and JAGS. For instance, if fixed cross-regression parameters are assumed, we do not need to assign a population distribution on these parameters. Instead, we can directly specify priors on these fixed parameters. When both person-specific auto- and cross-regression parameters are close to 0, the mlVAR model is reduced to a multilevel random-intercept model; we did not include this special case in the simulation study but instead refer the readers to existing resources (see, e.g., Bryk & Raudenbush, 1992; Candel, 2004; Snijders & Bosker, 2011).

Some possible future directions can be considered. First, the mlVAR model in the present study did not include time-varying covariates in the within-level model. Since time-varying covariates may play an important role in predicting the dynamics of variables of interest and the corresponding relationships, future work may consider fitting mlVAR-X models in different programs as well as handling missingness in covariates (Ji et al., 2020; Lee & Tang, 2006). Second, we considered only a small subset of factors when comparing the performance of the three software programs in fitting mlVAR models. Many other factors could also affect their performance, such as the extent of systematic between-level variations as compared to within-level variations and the nature of the VAR processes. Third, the mlVAR model considered in this study may be further extended to allow for evaluation of dynamics at the latent variable level (e.g., with Bayesian dynamic factor models; Chow, Tang, Yuan, Song, & Zhu, 2011; Zhang & Nesselroade, 2007). Future simulation studies should consider the effects of a more exhaustive list of factors in estimating Bayesian models of ILD of different levels of complexity.

Overall, we have demonstrated through a step-by-step coding example, an empirical example, and a simulation study how to fit a relatively complex variation of the mlVAR model in Stan, JAGS and Mplus. We provided some examples of research questions and novel substantive insights that may be gained using this model. In comparing and summarizing the strengths and limitations of each software program, we also pointed out some possible solutions, caveats, and unresolved challenges with respect to estimating the mlVAR model, particularly with regard to model and prior specifications, missing data inputs and handling techniques, computational efficiency and diagnostics. We hope that this tutorial can help lower some of the initial hurdles to fitting Bayesian mlVAR models, and provide some helpful exposure and guidelines to utilizing mlVAR and other related models to answer questions concerning intra-individual changes and inter-individual differences therein.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

Research reported in this publication was supported by the Intensive Longitudinal Health Behavior Cooperative Agreement Program funded by the National Institutes of Health under Award Number U24AA027684; and National Science Foundation grant IGE-1806874. Part of the computations for this research were performed on the Pennsylvania State University's Institute for Computational and Data Sciences' Roar supercomputer. We would like to thank Dr. Linhai Song for permission to use his lab server as additional computational resources.

References

Almeida DM, Piazza JR, & Stawski RS (2009). Interindividual differences and intraindividual variability in the cortisol awakening response: An examination of age and gender. Psychology and Aging, 24 (4), 819–827. [PubMed: 20025398]

Alvarez I, Niemi J, & Simpson M (2014). Bayesian inference for a covariance matrix. arXiv preprint arXiv:1408.4050.

Asparouhov T, Hamaker EL, & Muthén BO (2018). Dynamic structural equation models. Structural Equation Modeling: A Multidisciplinary Journal, 25(3), 359–388. Retrieved from 10.1080/10705511.2017.1406803 doi:10.1080/10705511.2017.1406803

Asparouhov T, & Muthén BO (2010). Plausible values for latent variables using Mplus. Unpublished. Accessed at: http://www.statmodel.com/download/Plausible.pdf.

Beck ED, & Jackson JJ (2020). Consistency and change in idiographic personality: A longitudinal ESM network study. Journal of Personality and Social Psychology, 118 (5), 1080. [PubMed: 31120288]

Betancourt M (2016). Diagnosing suboptimal cotangent disintegrations in Hamiltonian Monte Carlo. arXiv preprint arXiv:1604.00695.

Betancourt M (2017). A conceptual introduction to Hamiltonian Monte Carlo. arXiv preprint arXiv:1701.02434.

Bradburn NM (1969). The structure of psychological well-being. Aldine.

Bringmann LF, Ferrer E, Hamaker EL, Borsboom D, & Tuerlinckx F (2018). Modeling nonstationary emotion dynamics in dyads using a time-varying vector-autoregressive model. Multivariate Behavioral Research, 53(3), 293–314. [PubMed: 29505311]

Bringmann LF, Vissers N, Wichers M, Geschwind N, Kuppens P, Peeters F, … Tuerlinckx F (2013). A network approach to psychopathology: New insights into clinical longitudinal data. PloS one, 8 (4), e60188. [PubMed: 23593171]

Brose A, Schmiedek F, Koval P, & Kuppens P (2015). Emotional inertia contributes to depressive symptoms beyond perseverative thinking. Cognition and Emotion, 29, 527–538. doi: 10.1080/02699931.2014.916252 [PubMed: 24820350]

Bryk AS, & Raudenbush SW (1992). Hierarchical linear models: Applications and data analysis methods. Sage Publications, Inc.

Candel MJ (2004). Performance of empirical bayes estimators of random coefficients in multilevel analysis: Some results for the random intercept-only model. Statistica Neerlandica, 58(2), 197–219.

Canova F (2011). Methods for applied macroeconomic research. Princeton university press.

Carpenter B, Gelman A, Hoffman MD, Lee D, Goodrich B, Betancourt M, … Riddell A (2017). Stan: A probabilistic programming language. Journal of Statistical Software, 76(1).

Chow S-M, Haltigan JD, & Messinger DS (2010a). Dynamic infant-parent affect coupling during the face-to-face/still-face. Emotion, 10(1), 101–114. [PubMed: 20141307]

Chow S-M, Haltigan JD, & Messinger DS (2010b). Dynamic patterns of infant-parent interactions during face-to-face and still-face episodes. Emotion, 10(1), 101–114. [PubMed: 20141307]

Chow S-M, Tang N, Yuan Y, Song X, & Zhu H (2011). Bayesian estimation of semiparametric dynamic latent variable models using the Dirichlet process prior. British Journal of Mathematical and Statistical Psychology, 64 (1), 69–106. [PubMed: 21506946]

Chow S-M, & Zhang G (2013). Nonlinear regime-switching state-space (RSSS) models. Psychometrika: Application Reviews and Case Studies, 78(4), 740–768.

Cohen S, Kamarck T, & Mermelstein R (1983). A global measure of perceived stress. Journal of Health and Social Behavior, 24 (4), 385–396. [PubMed: 6668417]

Costa PT Jr., McCrae RR, & Dye DA (1991). Facet scales for agreeableness and conscientiousness: A revision of the NEO personality inventory. Personality and Individual Differences, 12(9), 887–898.

Daniels MJ, & Hogan JW (2008). Missing data in longitudinal studies: Strategies for Bayesian modeling and sensitivity analysis. London: Chapman and Hall.

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

Diener E, & Emmons RA (1984). The independence of positive and negative affect. Journal of Personality and Social Psychology, 47(5), 1105. [PubMed: 6520704]

Diggle P, & Kenward MG (1994). Informative drop-out in longitudinal data analysis. Journal of the Royal Statistical Society: Series C (Applied Statistics), 43(1), 49–73.

du Toit SH, & Browne MW (2007). Structural equation modeling of multivariate time series. Multivariate Behavioral Research, 42 (1), 67–101. [PubMed: 26821077]

Egloff B (1998). The independence of positive and negative affect depends on the affect measure. Personality and Individual Differences, 25(6), 1101–1109.

Emotions and Dynamic Systems Laboratory. (2010). The affective dynamics and individual differences (ADID) study: Developing non-stationary and network-based methods for modeling the perception and physiology of emotions. (Unpublished manual, University of North Carolina at Chapel Hill)

Epskamp S, Waldorp LJ, Mõttus R, & Borsboom D (2018). The Gaussian graphical model in cross-sectional and time-series data. Multivariate Behavioral Research, 53(4), 453–480. [PubMed: 29658809]

Fisher RA (1915). Frequency distribution of the values of the correlation coefficient in samples from an indefinitely large population. Biometrika, 10 (4), 507–521.

Gates KM, & Molenaar PC (2012). Group search algorithm recovers effective connectivity maps for individuals in homogeneous and heterogeneous samples. NeuroImage, 63 (1), 310–319. [PubMed: 22732562]

Gelman A, et al. (2006). Prior distributions for variance parameters in hierarchical models (comment on article by Browne and Draper). Bayesian Analysis, 1 (3), 515–534.

Gelman A, Rubin DB, et al. (1992). Inference from iterative simulation using multiple sequences. Statistical Science, 7(4), 457–472.

Goldstein MD, & Strube MJ (1994). Independence revisited: The relation between positive and negative affect in a naturalistic setting. Personality and Social Psychology Bulletin, 20 (1), 57–64.

Hallquist MN, & Wiley JF (2018). MplusAutomation: An R package for facilitating large-scale latent variable analyses in Mplus. Structural Equation Modeling: A Multidisciplinary Journal, 25(4), 621–638. [PubMed: 30083048]

Hamaker E, Asparouhov T, Brose A, Schmiedek F, & Muthén BO (2018). At the frontiers of modeling intensive longitudinal data: Dynamic structural equation models for the affective measurements from the COGITO study. Multivariate Behavioral Research, 53 (6), 820–841. [PubMed: 29624092]

Hamaker E, Asparouhov T, & Muthén BO (2017). Dynamic structural equation modeling of intensive longitudinal data using Mplus version 8.

Hamilton JD (1994). Time series analysis (Vol. 2). Princeton New Jersey.

Harvey AC (1990). Forecasting, structural time series models and the Kalman filter. Cambridge university press.

Hastings WK (1970). Monte Carlo sampling methods using Markov chains and their applications. Biometrika, 57(1), 97–109.

Hedeker D, & Mermelstein RJ (2007). Mixed-effects regression models with heterogeneous variance: Analyzing ecological momentary assessment (EMA) data of smoking. Modeling Contextual Effects in Longitudinal Studies, 183–206.

Hedeker D, Mermelstein RJ, Berbaum ML, & Campbell RT (2009). Modeling mood variation associated with smoking: An application of a heterogeneous mixed-effects model for analysis of ecological momentary assessment (EMA) data. Addiction, 104 (2), 297–307. [PubMed: 19149827]

Huber F, & Feldkircher M (2019). Adaptive shrinkage in bayesian vector autoregressive models. Journal of Business & Economic Statistics, 37(1), 27–39.

Ji L, Chen M, Oravecz Z, Cummings EM, Lu Z-H, & Chow S-M (2020). A Bayesian vector autoregressive model with nonignorable missingness in dependent variables and covariates: Development, evaluation, and application to family processes. Structural Equation Modeling: A Multidisciplinary Journal, 27(3), 442–467. [PubMed: 32601517]

Kernighan BW, & Ritchie DM (2006). The C programming language.

Korner-Nievergelt F, Roth T, Von Felten S, Guélat J, Almasi B, & Korner-Nievergelt P (2015). Bayesian data analysis in ecology using linear models with R, BUGS, and Stan. Academic Press.

Koval P, Kuppens P, Allen NB, & Sheeber L (2012). Getting stuck in depression: The roles of rumination and emotional inertia. Cognition and Emotion, 26, 1412–1427. [PubMed: 22671768]

Kruschke J (2014). Doing bayesian data analysis: A tutorial with R, JAGS, and Stan. Academic Press.

Kuppens P, Allen NB, & Sheeber LB (2010). Emotional inertia and psychological maladjustment. Psychological Science, 21 (7), 984–991. [PubMed: 20501521]

Lee S-Y, & Song X-Y (2004). Evaluation of the Bayesian and maximum likelihood approaches in analyzing structural equation models with small sample sizes. Multivariate Behavioral Research, 39 (4), 653–686. [PubMed: 26745462]

Lee S-Y, & Tang N-S (2006). Analysis of nonlinear structural equation models with nonignorable missing covariates and ordered categorical data. Statistica Sinica, 16, 1117–1141.

Lewandowski D, Kurowicka D, & Joe H (2009). Generating random correlation matrices based on vines and extended onion method. Journal of Multivariate Analysis, 100(9), 1989–2001. Retrieved from https://www.sciencedirect.com/science/article/pii/S0047259X09000876 doi: 10.1016/j.jmva.2009.04.008

Li Y, Ji L, Oravecz Z, Brick TR, Hunter MD, & Chow SM (2019). dynr.mi: An R Program for Multiple Imputation in Dynamic Modeling. International Journal of Computer, Electrical, Automation, Control and Information Engineering, 13(5), 302–311.

Lipsitz SR, & Ibrahim JG (1996). A conditional model for incomplete covariates in parametric regression models. Biometrika, 83(4), 916–922. Retrieved from http://www.jstor.org/stable/2337295.

Little RJA, & Rubin DB (1987). Statistical analysis with missing data. New York, N.Y.: Wiley.

Lucas RE, Le K, & Dyrenforth PS (2008). Explaining the extraversion/positive affect relation: Sociability cannot account for extraverts' greater happiness. Journal of Personality, 76(3), 385–414. [PubMed: 18399958]

Lütkepohl H (2005). Introduction to multiple time series analysis (2nd ed.). New York, NY: Springer-Verlag.

McElreath R (2018). Statistical rethinking: A Bayesian course with examples in R and Stan. Chapman and Hall/CRC.

Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, & Teller E (1953). Equation of state calculations by fast computing machines. The Journal of Chemical Physics, 21 (6), 1087–1092.

Monnahan CC, Thorson JT, & Branch TA (2017). Faster estimation of Bayesian models in ecology using Hamiltonian Monte Carlo. Methods in Ecology and Evolution, 8 (3), 339–348.

Muthén BO, Asparouhov T, Hunter AM, & Leuchter AF (2011). Growth modeling with nonignorable dropout: alternative analyses of the STAR*D antidepressant trial. Psychological Methods, 16(1), 17. [PubMed: 21381817]

Muthén LK, & Muthén BO (1998-2017). Mplus User's Guide. Eighth Edition. Los Angeles, CA.

Neal RM (2003). Slice sampling. Annals of Statistics, 31 (3), 705–741.

Neal RM (2011). MCMC using Hamiltonian dynamics. Handbook of Markov Chain Monte Carlo, 2(11), 2.

Oravecz Z, & Muth C (2018). Fitting growth curve models in the Bayesian framework. Psychonomic Bulletin and Review, 25(1), 235–255. [PubMed: 28493149]

Oravecz Z, Tuerlinckx F, & Vandekerckhove J (2011). A hierarchical latent stochastic differential equation model for affective dynamics. Psychological Methods, 16(4), 468. [PubMed: 21823796]

Oravecz Z, Tuerlinckx F, & Vandekerckhove J (2016). Bayesian data analysis with the bivariate hierarchical Ornstein-Uhlenbeck process model. Multivariate Behavioral Research, 51, 106–119. [PubMed: 26881960]

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, p. 1–10).

Plummer M, Stukalov A, & Denwood M (2019). rjags: Bayesian graphical models using MCMC. R package version 4–10.

Raghunathan T, Lepkowski J, van Hoewyk J, & Solenberger P (2001). A multivariate technique for multiply imputing missing values using a sequence of regression models. Survey Methodology, 27(1), 85–95.

Robert C, & Casella G (2013). Monte Carlo statistical methods. Springer Science & Business Media.

Schuurman NK (2018). Intensive longitudinal data analysis and DSEM. (Invited workshop at the Tilburg Experience Sampling Center, Tilburg, the Netherlands.)

Schuurman NK, Ferrer E, de Boer-Sonnenschein M, & Hamaker EL (2016). How to compare cross-lagged associations in a multilevel autoregressive model. Psychological Methods, 21 (2), 206. [PubMed: 27045851]

Schuurman NK, Grasman R, & Hamaker E (2016). A comparison of inverse-wishart prior specifications for covariance matrices in multilevel autoregressive models. Multivariate Behavioral Research, 51 (2-3), 185–206. [PubMed: 27028576]

Snijders TA, & Bosker RJ (2011). Multilevel analysis: An introduction to basic and advanced multilevel modeling. Sage.

Song H, & Zhang Z (2014). Analyzing multiple multivariate time series data using multilevel dynamic factor models. Multivariate Behavioral Research, 49(1), 67–77. Retrieved from 10.1080/00273171.2013.851018 doi: 10.1080/00273171.2013.851018 [PubMed: 26745674]

Sorensen T, & Vasishth S (2015). Bayesian linear mixed models using Stan: A tutorial for psychologists, linguists, and cognitive scientists. arXiv preprint arXiv:1506.06201.

Stan Development Team. (2018). Stan modeling language users guide and reference manual. Version 2.18.0. Retrieved from http://mc-stan.org/

Stan Development Team. (2020). RStan: the R interface to Stan. Retrieved from http://mc-stan.org/ (R package version 2.19.3)

Su Y-S, & Yajima M (2020). R2jags: Using R to run JAGS. R package version 0.6–1.

Tang N, Chow S-M, Ibrahim JG, & Zhu H (2017, Dec 01). Bayesian sensitivity analysis of a nonlinear dynamic factor analysis model with nonparametric prior and possible nonignorable missingness. Psychometrika, 82(4), 875–903. Retrieved from 10.1007/s11336-017-9587-4 doi: 10.1007/s11336-017-9587-4 [PubMed: 29030749]

van Buuren S (2007). Multiple imputation of discrete and continuous data by fully conditional specification. Statistical Methods and Medical Research, 16(3), 219–242.

van Buuren S, Brand J, Groothius-Oudshoorn C, & Rubin D (2006). Fully conditional specification in multivariate imputation. Journal of Statistical Computation and Simulation, 76 (12), 1049–1064.

Wabersich D, & Vandekerckhove J (2014). Extending JAGS: A tutorial on adding custom distributions to JAGS (with a diffusion model example). Behavior Research Methods, 46(1), 15–28. [PubMed: 23959766]

Wang LP, Hamaker EL, & Bergeman CS (2012). Investigating inter-individual differences in short-term intra-individual variability. Psychological Methods, 17(4), 567. [PubMed: 22924600]

Watson D (1988). Intraindividual and interindividual analyses of positive and negative affect: their relation to health complaints, perceived stress, and daily activities. Journal of Personality and Social Psychology, 54 (6), 1020. [PubMed: 3397861]

Watson D, & Clark LA (1997). Extraversion and its positive emotional core. In Handbook of personality psychology (pp. 767–793). Elsevier.

Watson D, Clark LA, & Tellegen A (1988). Development and validation of brief measures of positive and negative affect: The PANAS scales. Journal of Personality and Social Psychology, 54 (6), 1063. [PubMed: 3397865]

Wright AG, Gates KM, Arizmendi C, Lane ST, Woods WC, & Edershile EA (2019). Focusing personality assessment on the person: Modeling general, shared, and person specific processes in personality and psychopathology. Psychological Assessment, 31 (4), 502. [PubMed: 30920277]

You D, Hunter M, Chen M, & Chow S-M (2019). A diagnostic procedure for detecting outliers in linear state-space models. Multivariate Behavioral Research, 1–25. Retrieved from 10.1080/00273171.2019.1627659 doi: 10.1080/00273171.2019.1627659

Zhang Z, & Nesselroade JR (2007). Bayesian estimation of categorical dynamic factor models. Multivariate Behavioral Research, 42 (4), 729–756. doi: 10.1080/00273170701715998.

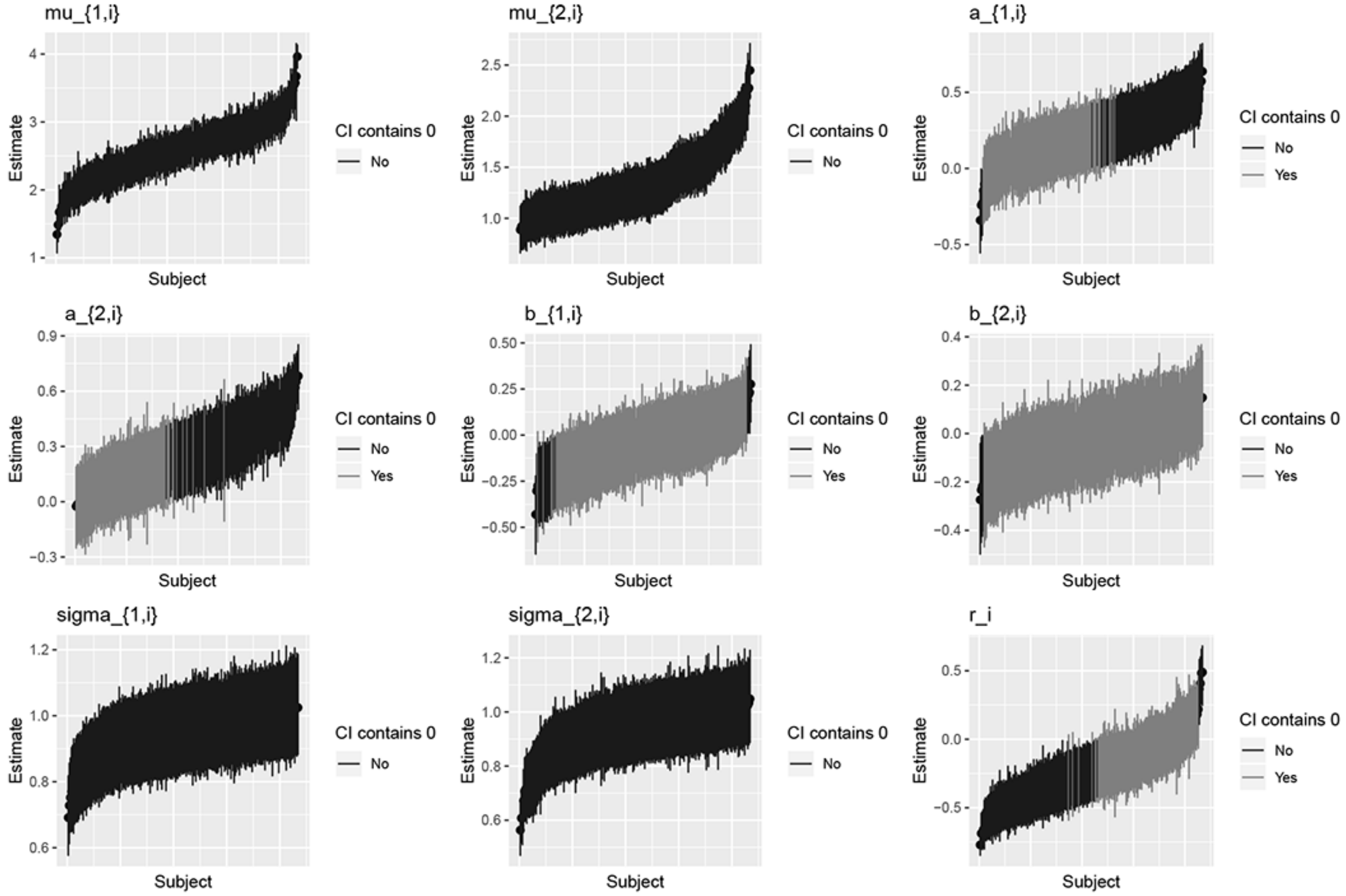

Figure 1.

Each plot represents the point estimates (displayed as points) and 95% credible intervals (displayed as error bars with different colors) of person-specific parameters for all participants, where point estimates are displayed in ascending order among all participants. In each plot, the black error bars represent credible intervals not containing 0 while the grey ones represent credible intervals containing 0.

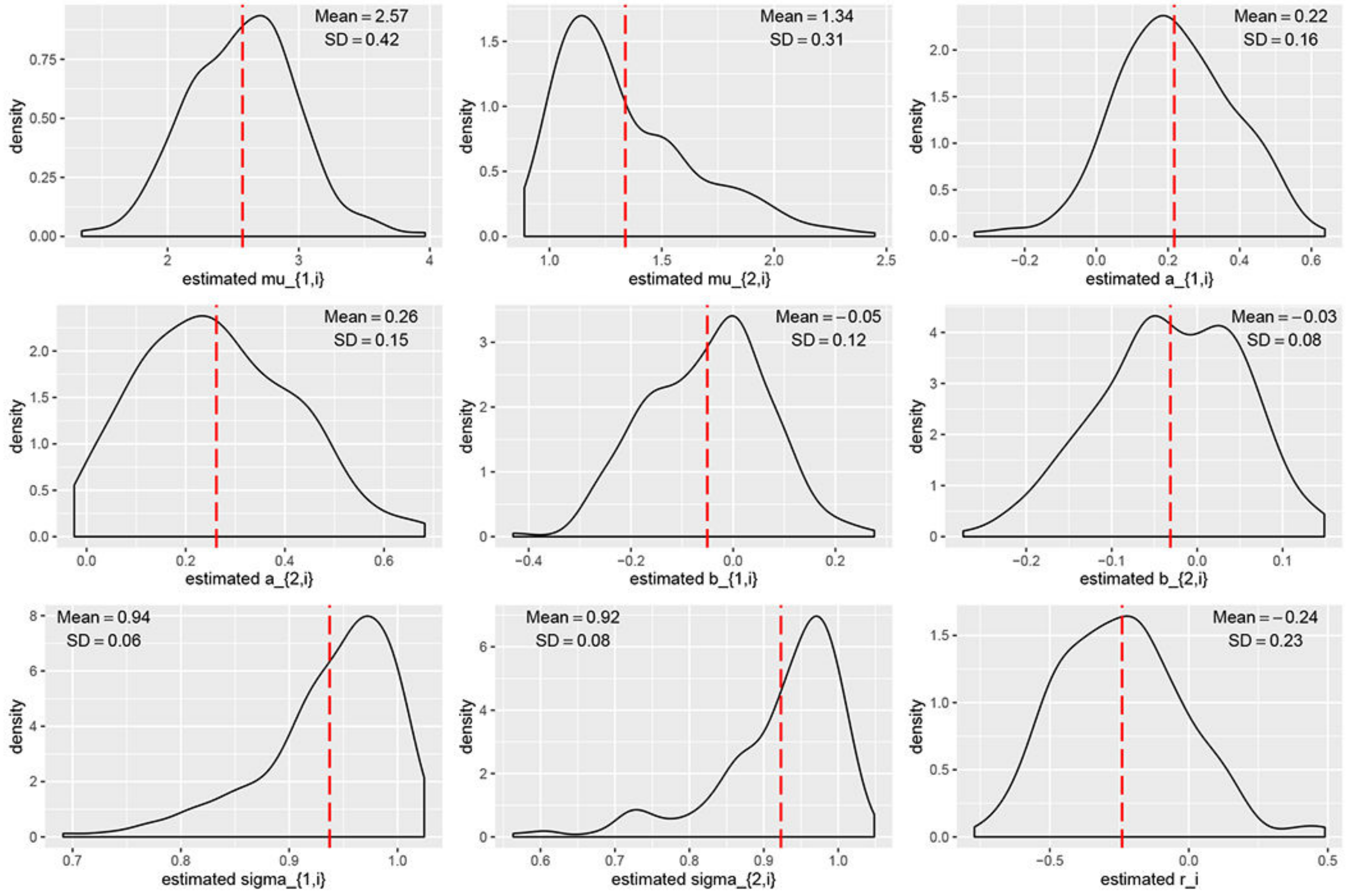

Figure 2.

Density plots of the point estimates of person-specific parameters across participants. The means (the red dashed lines) and standard deviations are printed in the plot.

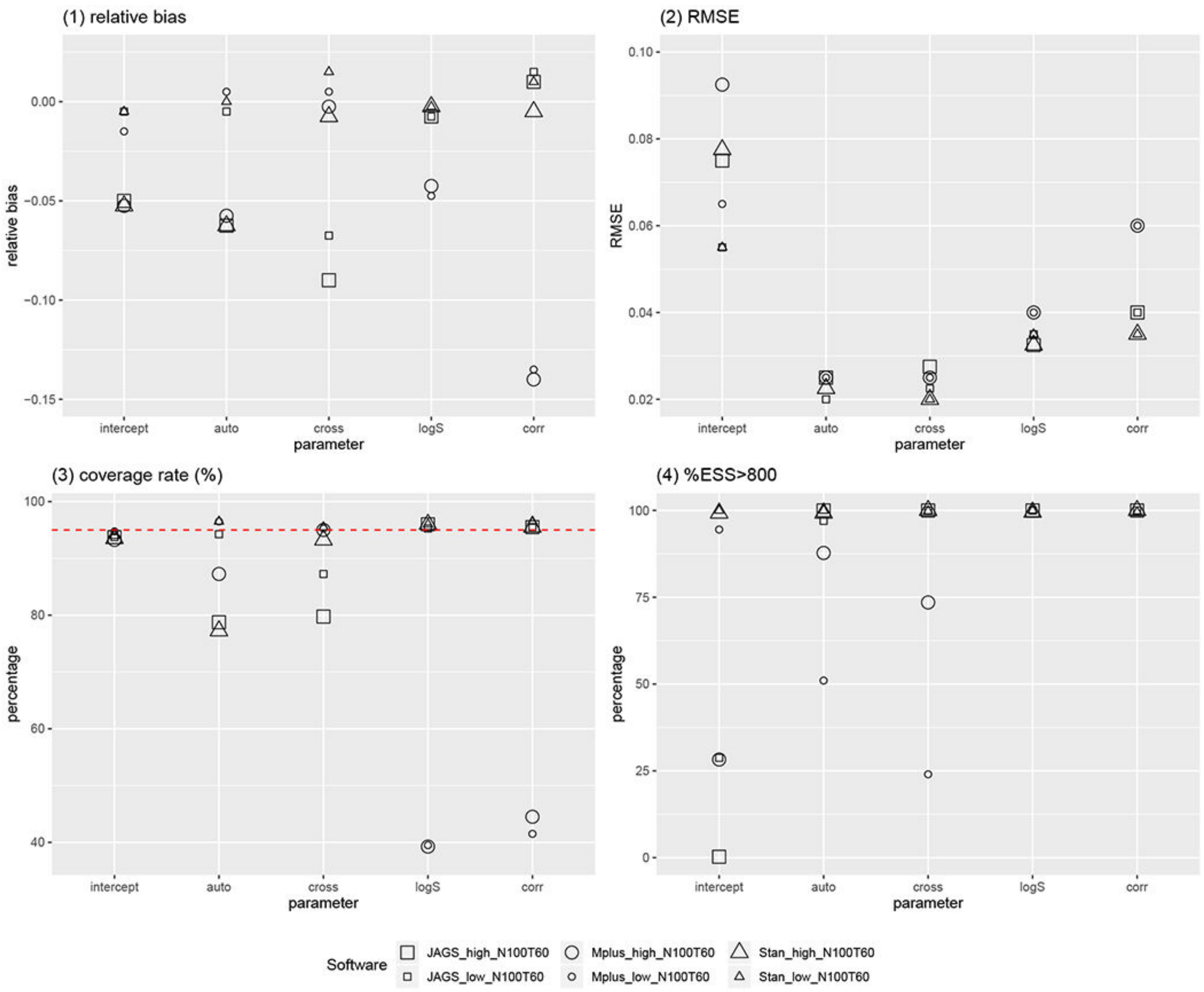

Figure 3.

Comparisons of (1) relative biases (2) RMSEs (3) coverage rates and (4) the percentages of replications with ESS greater than 800 for between-level regression coefficients across software programs under N = 100 and T = 60. The large (small) symbol represented the high-stability (low-stability) condition. The between-level regression coefficients were divided into five categories as described in the main text. Note that Mplus was not evaluated on ESS for standard deviation- and correlation-related parameters because the total number of posterior samples for these parameters was 1000 rather than 40000.

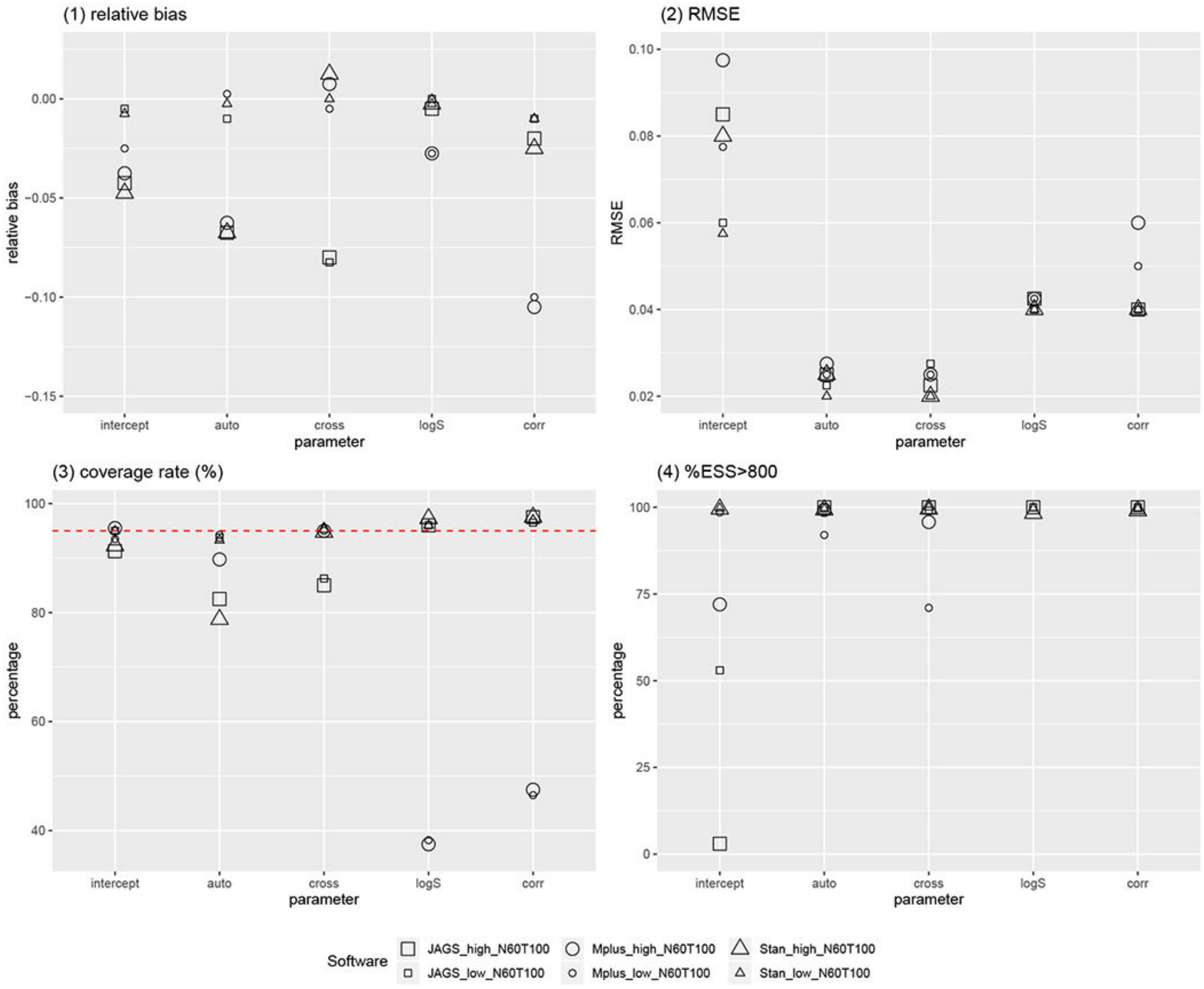

Figure 4.

Comparisons of (1) relative biases (2) RMSEs (3) coverage rates and (4) the percentages of replications with ESS greater than 800 for between-level regression coefficients across software programs under N = 60 and T = 100. The large (small) symbol represented the high-stability (low-stability) condition. The between-level regression coefficients were divided into five categories as described in the main text. As explained under Figure 3, Mplus was not evaluated on ESS.

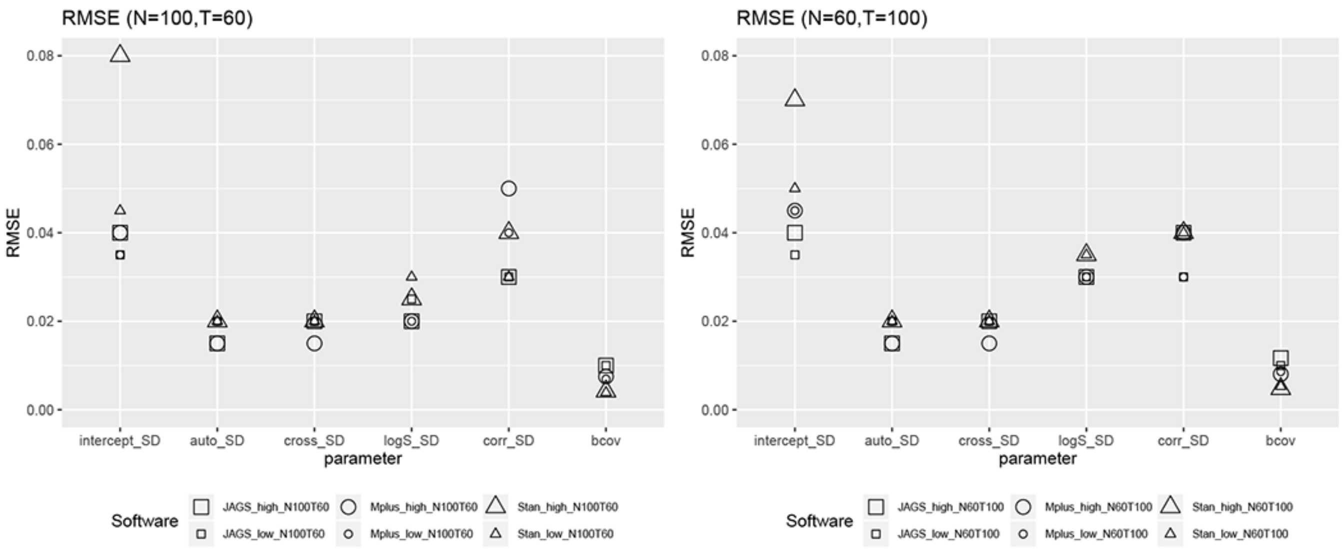

Figure 5.

Comparisons of RMSEs for random effect standard deviations and covariances across software programs under $N = 100$, $T = 60$ (left panel) and $N = 60$, $T = 100$ (right panel). The large (small) symbol represented the high-stability (low-stability) condition. Similar to between-level regression coefficients, the random effect standard deviations were divided into five categories. The “bcov” column represents the average of RMSEs for all random effect covariances.

Table 1

Parameter Estimates for Empirical Data

	Stan			JAGS			Mplus		
Par	**Est**	**SE**	**95% CI**	**Est**	**SE**	**95% CI**	**Est**	**SE**	**95% CI**
μ_1	2.57	0.03	[2.51 , 2.63]	2.57	0.03	[2.51 , 2.63]	2.58	0.03	[2.52 , 2.63]
$\alpha_{\mu 1}$	−0.03	0.03	[−0.09 , 0.04]	−0.02	0.03	[−0.08 , 0.04]	−0.02	0.03	[−0.08 , 0.04]
$\beta_{\mu 1}$	0.20	0.03	[0.14 , 0.26]	0.21	0.03	[0.15 , 0.27]	0.20	0.03	[0.14 , 0.26]
μ_2	1.34	0.02	[1.30 , 1.37]	1.34	0.02	[1.30 , 1.38]	1.36	0.02	[1.32 , 1.40]
$\alpha_{\mu 2}$	0.27	0.02	[0.23 , 0.30]	0.27	0.02	[0.23 , 0.31]	0.27	0.02	[0.23 , 0.31]
$\beta_{\mu 2}$	0.00	0.02	[−0.03 , 0.04]	0.00	0.02	[−0.04 , 0.04]	0.00	0.02	[−0.04 , 0.04]
a_1	0.22	0.02	[0.19 , 0.25]	0.22	0.02	[0.18 , 0.25]	0.22	0.02	[0.19 , 0.26]
α_{a1}	0.03	0.02	[0.00 , 0.06]	0.03	0.02	[−0.01 , 0.06]	0.02	0.02	[−0.01 , 0.06]
β_{a1}	0.04	0.02	[0.005, 0.07]	0.04	0.02	[0.002, 0.07]	0.04	0.02	[0.002, 0.07]
a_2	0.27	0.02	[0.24 , 0.30]	0.26	0.02	[0.23 , 0.30]	0.27	0.02	[0.24 , 0.31]
α_{a2}	0.05	0.01	[0.02 , 0.08]	0.04	0.02	[0.01 , 0.08]	0.04	0.02	[0.001, 0.07]
β_{a2}	0.00	0.01	[−0.03 , 0.03]	0.00	0.02	[−0.03 , 0.03]	−0.01	0.02	[−0.04 , 0.03]
b_1	−0.05	0.01	[−0.08 ,−0.02]	−0.05	0.02	[−0.08 ,−0.02]	−0.04	0.02	[−0.07 ,−0.01]
α_{b1}	−0.01	0.01	[−0.03 , 0.02]	−0.01	0.02	[−0.04 , 0.02]	0.00	0.02	[−0.04 , 0.03]
β_{b1}	−0.01	0.01	[−0.04 , 0.02]	−0.01	0.02	[−0.04 , 0.02]	−0.01	0.02	[−0.04 , 0.02]
b_2	−0.03	0.01	[−0.05 , 0.00]	−0.03	0.01	[−0.06 , 0.00]	−0.03	0.01	[−0.06 , 0.00]
α_{b2}	−0.01	0.01	[−0.03 , 0.01]	−0.01	0.01	[−0.04 , 0.02]	−0.01	0.01	[−0.04 , 0.02]
β_{b2}	−0.01	0.01	[−0.04 , 0.01]	−0.01	0.01	[−0.04 , 0.02]	−0.01	0.01	[−0.04 , 0.02]
$l\sigma_1$	−0.07	0.01	[−0.09 ,−0.06]	−0.07	0.01	[−0.10 ,−0.05]	−0.07	0.01	[−0.08 ,−0.05]
$\alpha_{l\sigma 1}$	0.00	0.01	[−0.02 , 0.01]	0.00	0.01	[−0.02 , 0.02]	0.00	0.00	[−0.01 , 0.01]
$\beta_{l\sigma 1}$	−0.02	0.01	[−0.04 , 0.00]	−0.02	0.01	[−0.04 , 0.00]	−0.01	0.00	[−0.02 , 0.00]
$l\sigma_2$	−0.09	0.01	[−0.11 ,−0.07]	−0.09	0.01	[−0.12 ,−0.07]	−0.07	0.01	[−0.08 ,−0.06]
$\alpha_{l\sigma 2}$	0.01	0.01	[−0.01 , 0.03]	0.01	0.01	[−0.02 , 0.03]	0.00	0.00	[−0.01 , 0.01]
$\beta_{l\sigma 2}$	0.00	0.01	[−0.02 , 0.01]	−0.01	0.01	[−0.03 , 0.02]	0.00	0.00	[−0.01 , 0.00]
z	−0.26	0.02	[−0.31 ,−0.22]	−0.26	0.02	[−0.31 ,−0.22]	−0.24	0.01	[−0.26 ,−0.22]
α_z	−0.03	0.02	[−0.08 , 0.01]	−0.03	0.02	[−0.08 , 0.02]	−0.02	0.01	[−0.04 , 0.00]
β_z	0.02	0.02	[−0.02 , 0.07]	0.02	0.02	[−0.02 , 0.07]	0.02	0.01	[0.00 , 0.04]
ψ_1	0.40	0.02	[0.35 , 0.45]	0.40	0.02	[0.35 , 0.45]	0.38	0.02	[0.34 , 0.43]
ψ_2	0.16	0.02	[0.12 , 0.20]	0.21	0.02	[0.18 , 0.24]	0.20	0.02	[0.17 , 0.23]
ψ_3	0.18	0.01	[0.16 , 0.21]	0.21	0.01	[0.19 , 0.24]	0.21	0.01	[0.19 , 0.24]
ψ_4	0.16	0.01	[0.14 , 0.19]	0.20	0.01	[0.18 , 0.23]	0.21	0.01	[0.18 , 0.24]
ψ_5	0.13	0.01	[0.11 , 0.16]	0.18	0.01	[0.15 , 0.20]	0.18	0.01	[0.15 , 0.20]
ψ_6	0.08	0.01	[0.05 , 0.11]	0.15	0.01	[0.13 , 0.17]	0.15	0.01	[0.13 , 0.17]

	Stan			JAGS			Mplus		
Par	**Est**	**SE**	**95% CI**	**Est**	**SE**	**95% CI**	**Est**	**SE**	**95% CI**
ψ_{7}	0.05	0.01	[0.04 , 0.07]	0.12	0.01	[0.11 , 0.14]	0.08	0.01	[0.07 , 0.10]
ψ_{8}	0.10	0.01	[0.08 , 0.12]	0.15	0.01	[0.13 , 0.17]	0.08	0.01	[0.07 , 0.09]
ψ_{9}	0.29	0.02	[0.26 , 0.33]	0.30	0.02	[0.26 , 0.33]	0.26	0.01	[0.23 , 0.28]
ψ_{12}	0.02	0.01	[0.01 , 0.04]	0.02	0.01	[0.01 , 0.04]	0.02	0.01	[0.01 , 0.04]
ψ_{34}	0.01	0.00	[0.01 , 0.02]	0.01	0.00	[0.004, 0.02]	0.01	0.00	[0.004, 0.02]
ψ_{37}	−0.01	0.00	[−0.01 ,−0.004]	−0.01	0.00	[−0.01 ,−0.004]	−0.01	0.00	[−0.01 ,−0.005]
ψ_{47}	−0.01	0.00	[−0.01 ,−0.002]	−0.01	0.00	[−0.01 ,−0.001]	−0.01	0.00	[−0.01 ,−0.003]
ψ_{48}	−0.01	0.00	[−0.02 ,−0.01]	−0.01	0.00	[−0.02 ,−0.01]	−0.01	0.00	[−0.02 ,−0.01]
ψ_{56}	0.01	0.00	[0.004, 0.01]	0.01	0.00	[0.003, 0.01]	0.01	0.00	[0.003, 0.01]

Table 2

Comparisons across Stan, JAGS, and Mplus

Features	**Stan**	**JAGS**	**Mplus**
Accessibility	Open-source	Open-source	Proprietary
R Packages	`rstan`	`rjags, R2jags, runjags`	`MplusAutomation`
Model Scripts	Intense	Detailed	Concise
Model Specifications	Flexible	Flexible	Limited (e.g., constraints on random innovation correlations)
Prior Specifications	Flexible	Flexible	Limited
Missing Data Inputs	Missing flags are needed, and they need to be replaced by a vector parameter in the model script	Missing flags are not needed; users cannot specify multivariate distributions for partially observed nodes	Missing flags are needed
Missing Data Handling Techniques	Fully Bayesian approach; users need to write their own code to specify the hypothesized missing data models	Fully Bayesian approach; users need to write their own code to specify the hypothesized missing data models	Automatic multiple imputation; more restricted missing data modeling options
MCMC Algorithms	Hamiltonian Monte Carlo (HMC)	Gibbs sampling, Slice sampling	Metropolis-Hastings (MH)
Computational Time	High	Low	Low
Diagnostics	Provide warning messages about pathologies	Limited warning messages	Limited warning messages