# Bayesian Estimation of Multilevel Structural Equation Models: Prior Specification for Random Effects Variances

Jesse Vonk 5235960

Bachelor in Applied Mathematics

TU Delft 11-7-2025

June 2025

# Abstract

Multilevel Structural Equation Modeling (MLSEM) is a framework for statistical modeling of hierarchical data and variables that are not observed directly, so called latent variables. Hierarchical data, or nested data, is a data structure that is common in psychology, organizational- and educational sciences. An example of hierarchical data is measurements made on students, such as grades, sampled from multiple schools. In this example, students are said to be nested within schools. The school that the student attends can be a causal factor of the grades that the student gets, and students from the same school thus share a common factor, namely the school. This causes the data collected from students, that attend the same school, to be dependent, which violates an assumption common in statistical modeling, namely that the samples need to be independent. This violation of the assumption of independence thus needs to be dealt with in a special manner, which leads to multilevel modeling, where the sampled data is split into a component caused by the student (individual) and a component that is caused by the school (group).

MLSEM combines the multilevel modeling framework with the Structural Equation Modeling (SEM) framework. The SEM framework models relationships between variables that can not be observed directly, but are theorized to be related to variables that can be observed directly. An example is academic performance related to grades. Academic performance cannot be observed directly, and as such is called a latent variable. However, it isrelated to variables that can be observed directly, namely test scores. Academic performance can be related to multiple test scores and through the SEM framework related to other latent variables, such as stress.

MLSEM combines these frameworks by accounting for hierarchical data and the use of latent variables. It does so by decomposing the data into within- (e.g. student) and between- (e.g. school) components and specifying a SEM model for both components.

In this thesis, in chapter 2 an overview of the mathematical foundations underlying Multilevel Structural Equation Modeling (MLSEM) is given. This chapter explains the basic components of MLSEMs, starting from the within-between decomposition of the variance in equation 2.1.1. The independent components, resulting from the within-between decomposition, are both described by a single level SEM model. Together, these SEM models describe the multilevel structure and the relationships among latent variables and between latent- and observed variables. In chapter 3 the foundations of Bayesian estimation, including an illustration of the influence of priors, are given and extended to the Bayesian estimation of MLSEM. In chapter 4 a simulation study comparing three priors for random effects variances, available in the R package Blavaan, is done. This simulation study shows that the gamma distribution specified on the standard deviation of random effects, rather than on the variance directly, performs best especially when the true variance parameter is small. In chap-

ter 5, the results from the simulation study are applied to a real-life dataset from the PRIME research program, which investigates the relationship between cognitive load and perceived mental effort.

# Contents

# 1 Introduction

Bayesian multilevel structural equation modeling (MLSEM) is an important statistical model to study relationships between latent variables with hierarchically structured data, as often encountered in educational- and organizational settings. Bayesian estimation of MLSEMs allow researchers to use prior information to get better estimates, especially when working with small sample sizes. Over the past two decades significant progress has been made in the development of Bayesian estimation methods for MLSEMs. Researchers have investigated robust prior distributions to improve model stability and estimation accuracy. Prior specifications have been investigated to improve the robustness of doubly latent categorical multilevel models [8], addressing challenges associated with small sample sizes and complex model structures (Zitmann et al.,2020) [1]. The choice of prior distribution on random effects variances has been an important component of Bayesian MLSEM research. Studies have shown that slightly informative priors can stabilize estimators and can reduce the mean squared error (MSE) of between-group slope estimates. Two methods have been proposed: specifying informative priors directly for the slopes and for the between-group variance of predictor variables. Both methods have shown effectiveness in improving estimation accuracy, especially in small sample contexts (Zitmann et al.,2020) [1]. A recent advance has been the integration of Bayesian estimation of MLSEMs in statistical software. For example, the R package 'Blavaan' enables Bayesian estimation of MLSEMs using Jags. 'Blavaan' also provides researchers with accessible tools for model fitting and result summarization (Merkle & Rosseel, 2018)[17].
Despite these advances, several areas require further investigation.
For example, there is a great need for guidelines regarding prior specification for random effects variances to ensure model stability and estimation accuracy across research contexts.
Additionally, enhancing user-friendliness and computational efficiency of Bayesian estimation tools can encourage wider adoption among applied researchers, facilitating the implementation of these advanced modeling techniques.
Because of the need for guidelines for prior specification and the influence priors can have on estimation accuracy and model stability, analysis of these topics is promising. Furthermore, since user-friendly packages, like 'Blavaan', are important to the broader scientific community, analysis of prior specification and guidelines regarding priors within these packages can offer value to researchers from diverse disciplines.
This thesis will thus address the following objectives:

- Explore available prior distributions within 'Blavaan' for MLSEMs

- Provide guidelines for prior specification for random effects variances within 'Blavaan'

## 1.1 Thesis Outline

To this end, this thesis will:

- Introduce the key concepts of Bayesian MLSEM
- Introduce the mathematical foundations of Bayesian estimation in MLSEM
- Perform a simulation study to examine the effect of prior specification on estimation accuracy
- Apply the approach to real-world data from a psychology research study

# 2 MLSEM

## 2.1 Multilevel SEM

Multilevel modeling is a branch of statistics that addresses hierarchically structured or nested data. In hierarchical data - also known as clustered or nested data - data points are best conceptualized as being nested within groups or clusters. A good example of hierarchical data is students nested within classes. The measurements we associate with students, for example test scores, are conceptualized to be nested within a cluster, which in this case is the class the student belongs to. Multilevel models (MLM) also introduce group-level variables - measurements that are associated with the group rather than with individuals, for example teacher experience for a specific class. Multilevel models describe the nested structure of the data with levels. In this example the students belong to the lowest level, level 1, where as the classes belong to level 2. If we were to sample classes - and students within these classes - from multiple schools, we introduce a new level, level 3, to which the schools belong. These samples within a common cluster or group are often not independent, which is something that needs to be addressed, since many models such as ordinary linear regression assume complete independence of the samples. Failing to account for the nested data structure can lead to biased estimates (Depaoli, 2015) [6]. In order to address the group dependencies, multilevel models use group-level effects or components to model variability that is shared by individuals in the same group. An example of a group-level component, that will be used in this thesis, is a random intercept, which models variability in group means. Equation 2.1.1 is a simple example of a random intercept model.

$$Y_{ij} = \mu_Y + U_j + \varepsilon_{ij} \,, \tag{2.1.1}$$

$$U_j \sim \mathcal{N}(0, \Sigma_U) \,, \tag{2.1.2}$$

$$\varepsilon_{ij} \sim \mathcal{N}(0, \Sigma_\varepsilon) \,, \tag{2.1.3}$$

where $i = 1, \cdots, N_j$ denotes the individual nested within group $j = 1, \cdots, J$. Here the term $\mu_Y$ is the grand mean of $Y_{ij}$, $U_j$ is the group-level variability component and $\varepsilon_{ij}$ is the within-group variability component. In this example, we denote the variance of $U_j$, which is known as a random effects variance, with $\Sigma_U$ and the variance of $\varepsilon_{ij}$ with $\Sigma_\varepsilon$.
We could alternatively write:

$$\begin{aligned} Y_{ij} &= \mu_j + \varepsilon_{ij} \,, \\ \mu_j &= \mu_Y + U_j \,, \end{aligned} \tag{2.1.4}$$

such that:

$$\begin{aligned} Y_{ij} &\sim \mathcal{N}(\mu_j \,, \Sigma_\varepsilon) \,, \\ \mu_j &\sim \mathcal{N}(\mu_Y \,, \Sigma_U) \,, \end{aligned} \tag{2.1.5}$$

i.e. $Y_{ij}$ is distributed with mean $\mu_j$, where $\mu_j$ itself is a random variable with mean $\mu_Y$, hence the name random intercept.
Equation 2.1.1 will be used later in this chapter to decompose observed data into within and between components and plays an important role in MLSEM. In MLM, it's possible to also vary regression slopes between groups, but this is beyond the scope of this thesis and won't be used in the MLSEM models. For a more thorough overview of multilevel models and multilevel relationships, I refer the reader to Hox (2010)[11].

The Structural Equation Modeling (SEM) framework makes use of two types of variables, observed variables (OVs) and latent variables (LVs). The latter represents hypothetical (or theoretical) constructs that cannot be measured directly. Latent variables are defined by a set of observed variables that represent the construct; this is made explicit in the so called measurement model of the SEM model. SEM are widely applied in the social and behavioral sciences. An example of this are the PRIME Research questionnaire items that measure the various types of cognitive load. The measurement model, as we will see later in this chapter, incorporates residual terms that represent measurement error of the latent construct as a consequence of representing them with observed variables. The SEM modeling framework furthermore allows the specification of relationships between the latent variables. This enables researchers to study effects between variables that cannot be measured directly. These relationships are made explicit in the so called structural model of the SEM model. Thus SEM models enable the study of relationships between variables that can not be measured directly, while accounting for possible measurement error that occurs when we represent these latent constructs with observed variables that measure these constructs. Latent variables and the observed variables that measure them often represent theoretical constructs. For example, the PRIME research represents cognitive load with latent variables and uses questionnaires that measure cognitive load according to Cognitive Load Theory (see chapter 5).
The multilevel SEM (MLSEM) framework seeks to address both the hierarchical structure of the data and the study of latent constructs defined through observed variables. The general MLSEM framework allows the investigation of moderation effects between latent variables (through latent variable interactions) as well as group differences as modeled by random intercepts. The general framework has also been extended to nonlinear MLSEM models that allow the use of nonlinear relationships between latent variables. In this thesis we will be focusing on a two-level random intercept SEM model. In a two-level random intercept SEM model, there are no latent variables interactions or random coefficients (regression coefficients and factor loadings that vary between groups); rather group differences will be modeled solely by random intercepts. The reason this thesis will focus on the two-level random intercept SEM model is the specific needs posed by the research questions of the PRIME Research.
This chapter will lay out the basics of the two-level MLSEM model, covering model notation, the likelihood function, a derivation of model implied covariance matrix, model specification and identification.

## 2.2 Two-level SEM

The two-level random intercept SEM model can be specified by defining a measurement- and structural model at each level of the data (Muthén, 1994 & Depaoli, 2021)[7][16] . Each model describes the variability, i.e. variance around the (group-) mean, at that level, i.e. within-level and between-level variability.

**Observed Variables** First, we define the observed variables and the individuals, nested within groups, they belong to.
To this end, let, for individual $i = 1, \cdots, N_j$ : nested within group $j = 1, \cdots, J$ (level 2):

- $Y_{ij}$ be a $k$-variate vector containing the observed endogenous variables,
- $X_{ij}$ be a $p$-variate vector containing the observed exogenous variables,

where $J$ denotes the number of groups and $N_j$ denotes the number of individuals within group $j$.

Endogenous (dependent) refers to the fact that variability in $Y_{ij}$ will be explained by other variables in the model, namely exogenous (independent) variables $X_{ij}$. The designation of variables as either endogenous or exogenous is due to the use of the variables in the model and is not an intrinsic property of the variables themselves. Variables that are designated as exogenous in one model can be endogenous in another model.

**Within-Between Decomposition** The observed variables, $Y_{ij}$ and $X_{ij}$, are decomposed into independent within and between components using the random intercept model from 2.1.1:

$$\begin{aligned} Y_{ij} &= \mu_Y + Y_{ij}^W + Y_j^B \ , \\ X_{ij} &= \mu_X + X_{ij}^W + X_j^B \ . \end{aligned} \tag{2.2.1}$$

where

- $Y_j^B$, $X_j^B$ represent the between group variability, i.e. represents the variability between group-means
- $Y_{ij}^W$, $X_{ij}^W$ represent the within group variability, i.e. represents the variability around group-means
- $\mu_Y$ and $\mu_X$ are the grand means of $Y_{ij}$ and $X_{ij}$ , respectively.

The within and between components, for both $Y_{ij}$ and $X_{ij}$, have the property that they are orthogonal and additive (Searle, Casella, & McCulloch, 1992) [18], i.e. we can decompose the covariance matrices of the observed variables, denoted $\Sigma^T_Y$, $\Sigma^T_X$ and $\Sigma^T_{XY}$, as follows:

$$\Sigma^T_Y = \Sigma^W_Y + \Sigma^B_Y \,,$$

$$\Sigma^T_X = \Sigma^W_X + \Sigma^B_X \,, \tag{2.2.2}$$

$$\Sigma^T_{XY} = \Sigma^W_{XY} + \Sigma^B_{XY} \,.$$

For simplicity of notation, we define the covariances of $(Y^W_{ij}, X^W_{ij})^T$ and $(Y^B_j, X^B_j)^T$ as follows:

$$\Sigma^W = \begin{pmatrix} \Sigma^W_Y & \Sigma^W_{YX} \\ \Sigma^W_{XY} & \Sigma^W_X \end{pmatrix} \tag{2.2.3}$$

$$\Sigma^B = \begin{pmatrix} \Sigma^B_Y & \Sigma^B_{YX} \\ \Sigma^B_{XY} & \Sigma^B_X \end{pmatrix} \tag{2.2.4}$$

The SEM modeling framework conventionally assumes that the within and between components are multivariate normal distributed (Bollen, 1989)[2]:

$$(Y^W_{ij}, X^W_{ij})^T \sim \mathcal{N}(0, \Sigma^W) \,, \tag{2.2.5}$$

$$(Y^B_j, X^B_j)^T \sim \mathcal{N}(0, \Sigma^B) \,, \tag{2.2.6}$$

such that the observed variables are also multivariate normal distributed:

$$(Y_{ij}, X_{ij})^T \sim \mathcal{N}(\,(\mu_Y, \mu_X)^T \,, \Sigma^W + \Sigma^B) \,, \tag{2.2.7}$$

**Model Equations** Now that we have decomposed the data into within and between components, we describe them with level 1 and level 2 structural- and measurement models, respectively.

**Level 1**

$$Y^W_{ij} = \Lambda^W_Y \eta^W_{ij} + \varepsilon^W_{ij} \,, \tag{2.2.8}$$
$$X^W_{ij} = \Lambda^W_X \xi^B_{ij} + \delta^W_{ij} \,, \tag{2.2.9}$$
$$\eta^W_{ij} = B^W \eta^W_{ij} + \Gamma^W \xi^W_{ij} + \zeta^W_{ij} \,, \tag{2.2.10}$$

**Level 2**

$$Y^B_j = \Lambda^B_Y \eta^B_j + \varepsilon^B_j \,, \tag{2.2.11}$$
$$X^B_j = \Lambda^B_X \xi^B_j + \delta^B_j \,, \tag{2.2.12}$$
$$\eta^B_j = B^B \eta^B_j + \Gamma^B \xi^B_j + \zeta^B_j \,, \tag{2.2.13}$$

where we assume the following:

- $\varepsilon_{ij}^{W} \sim \mathcal{N}(0, \Theta_{\varepsilon}^{W})$ and $\varepsilon_{j}^{B} \sim \mathcal{N}(0, \Theta_{\varepsilon}^{B})$ ,
- $\delta_{ij}^{W} \sim \mathcal{N}(0, \Theta_{\delta}^{W})$ and $\delta_{j}^{B} \sim \mathcal{N}(0, \Theta_{\delta}^{B})$ ,
- $\zeta_{ij}^{W} \sim \mathcal{N}(0, \Psi^{W})$ and $\zeta_{j}^{B} \sim \mathcal{N}(0, \Psi^{B})$ ,
- $\xi_{ij}^{W} \sim \mathcal{N}(0, \Phi^{W})$ and $\xi_{j}^{B} \sim \mathcal{N}(0, \Phi^{B})$ ,
- $\mathrm{Cov}(\zeta_{ij}^{W}, \xi_{ij}^{W}) = 0$ and $\mathrm{Cov}(\zeta_{j}^{B}, \xi_{j}^{B}) = 0$ .

For a more extensive overview of the variables and parameters used in the two-level random intercept MLSEM model, see tables 1 and 2, respectively.

### Interpretation

**Hierarchical Structure** The model reflects that the data is nested by specifying individuals $i$ (level 1) within clusters $j$ (level 2), for example multiple questionnaires (level 1) per student (level 2). The multilevel structure allows for group dependencies to be taken into account: questionnaires filled in by the same student tend to be similar (see chapter 5). Furthermore, by partitioning the variance into within-components (level 1) and between-components, MLSEM allows researchers to test hypotheses for both within-level and between-level processes, for example, by testing hypotheses on the between-level regression coefficients to test contextual effects or by testing hypotheses on within-level coefficients to test individual effects (Hox, J., 2010)[11].

**Random Intercepts and Between-group Variability** Latent variables and between-components (e.g., $\eta_{j}^{B}$, $X_{j}^{B}$) model between group variability (Hox, J., 2010)[11]. The between-components of the observed variables, $Y_{j}^{B}$ and $X_{j}^{B}$, can be understood as random intercepts, as in equation 2.1.1. These latent variables represent group-level effects that are shared by all individuals (level 1) nested within the same group (level 2). Latent variables at the between level can be used to model contextual effects. For example, they can be used to explain why certain students (level 2) score higher overall on certain questionnaire items after accounting for individual (level 1) effects.

**Latent- and Observed Variables** At each level we include **exogenous latent variables** $\xi$ : these represent unobserved causes that influence endogenous outcomes; **endogenous latent variables** $\eta$ : these represent unobserved traits or outcomes that are influenced by exogenous or other endogenous variables; **observed indicator variables** $X$ , $Y$ : these variables are observed and reflect the latent constructs or variables used in the model. We include a **measurement model** at each level, which describes how latent variables are measured by the observed indicator variables, through the factor loadings (e.g. $\Lambda_{Y}^{W}$ ), and the measurement error (e.g. $\Theta_{\delta}^{B}$ ). We can, for example, measure cognitive load (latent) using questionnaire items. By specifying a measurement model at

each level we can measure within and between-level latent variables. Within-level latent variables are measured by the within-components of the data (e.g., $Y_{ij}^{W} \leftarrow \eta_{ij}^{W}$, through $\Lambda_Y^W$). Similarly, between-level latent variables are measured by between-components (e.g., $X_j^B \leftarrow \xi_j^B$, through $\Lambda_X^B$).
Note that the dimensions of the within- and between-level latent variables may be different. That is, the dimension of $\eta_{ij}^W$ does not have to equal the dimension of $\eta_j^B$. The same goes for $\xi_{ij}^W$ and $\xi_j^W$.
For further information, I refer the reader to Depaoli (2021) [7], who gives an example of a 3-level model with two LVs at level 1 and 1 LV at both level 2 and 3.

**Model Assumptions**

$$\begin{aligned} Y_{ij}^W &\sim \mathcal{N}(0, \Sigma_Y^W)\,, \\ X_{ij}^W &\sim \mathcal{N}(0, \Sigma_X^W)\,. \end{aligned}$$

$$\begin{aligned} Y_j^B &\sim \mathcal{N}(0, \Sigma_Y^B)\,, \\ X_j^B &\sim \mathcal{N}(0, \Sigma_X^B)\,. \end{aligned}$$

$$\begin{aligned} \eta_{ij}^W &\sim \mathcal{N}(0, \Sigma_\eta^W)\,, \\ \eta_j^B &\sim \mathcal{N}(0, \Sigma_\eta^B)\,. \end{aligned}$$

$$\begin{aligned} \xi_{ij}^W &\sim \mathcal{N}(0, \Phi_W)\,, \\ \xi_j^B &\sim \mathcal{N}(0, \Phi_B)\,. \end{aligned}$$

$$\begin{aligned} \zeta_{ij}^W &\sim \mathcal{N}(0, \Psi^W)\,, \\ \zeta_j^B &\sim \mathcal{N}(0, \Psi^B)\,. \end{aligned}$$

$$\begin{aligned} \varepsilon_{ij}^W &\sim \mathcal{N}(0, \Theta_\varepsilon^W)\,, \\ \varepsilon_j^B &\sim \mathcal{N}(0, \Theta_\varepsilon^B)\,. \end{aligned}$$

$$\begin{aligned} \delta_{ij}^W &\sim \mathcal{N}(0, \Theta_\delta^W)\,, \\ \delta_j^B &\sim \mathcal{N}(0, \Theta_\delta^B)\,. \end{aligned}$$

| Symbol | Name | Dimension | Definition | Level |
|---|---|---|---|---|
| | | **(Decomposed) Observed Variables (OV)** | | |
| $X_{ij}^W$ | X Within | $p \times 1$ | Within component of exog. OV | Level 1 |
| $X_j^B$ | X Between | $p \times 1$ | Between component of exog. OV | Level 2 |
| $Y_{ij}^W$ | Y Within | $k \times 1$ | Within component of endog. OV | Level 1 |
| $\mu_j^Y$ | Y Between | $k \times 1$ | Between component of endog. OV | Level 2 |
| | | **Latent Variables (LV)** | | |
| $\eta_{ij}^W$ | Eta Within | $m_1 \times 1$ | Within component of endog. LV | Level 1 |
| $\eta_j^B$ | Eta Between | $m_2 \times 1$ | Between component of endog. LV | Level 2 |
| $\xi_{ij}^W$ | Xi Within | $n_1 \times 1$ | Within component of exog. LV | Level 1 |
| $\xi_j^B$ | Xi Between | $n_2 \times 1$ | Between component of exog. LV | Level 2 |
| | | **Residuals and Measurement Error** | | |
| $\delta_{ij}^W$ | Delta Within | $p \times 1$ | Within exog. measurement error | Level 1 |
| $\delta_j^B$ | Delta Between | $p \times 1$ | Between exog. measurement error | Level 2 |
| $\varepsilon_{ij}^W$ | Epsilon Within | $k \times 1$ | Within endog. measurement error | Level 1 |
| $\varepsilon_j^B$ | Epsilon Between | $k \times 1$ | Between endog. measurement error | Level 2 |
| | | **Structural Residual** | | |
| $\zeta_{ij}^W$ | Zeta Within | $m_1 \times 1$ | Within structural residual | Level 1 |
| $\zeta_j^B$ | Zeta Between | $m_2 \times 1$ | Between structural residual | Level 2 |

Table 1: Overview of the variables used in a two-level random intercept MLSEM model. The table displays the symbol used, the name, the dimension, the definition and level for each variable

**Model Variables**

| Symbol | Dimension | Definition | Level |
|---|---|---|---|
| | | **Factor Loadings** | |
| $\Lambda_X^W$ | $p \times n_1$ | Relates $X_{ij}^W$ to $\xi_{ij}^W$ | Level 1 |
| $\Lambda_X^B$ | $p \times n_2$ | Relates $X_j^B$ to $\xi_j^B$ | Level 2 |
| $\Lambda_Y^W$ | $k \times m_1$ | Relates $Y_{ij}^W$ to $\eta_{ij}^W$ | Level 1 |
| $\Lambda_Y^W$ | $k \times m_2$ | Relates $Y_j^B$ to $\eta_j^B$ | Level 2 |
| | | **Structural Regression Slopes** | |
| $B^W$ | $m_1 \times m_1$ | Within-level relationships between endog. LVs: $\eta_{ij}^W$ | Level 1 |
| $B^B$ | $m_2 \times m_2$ | Between-level relationships between endog. LVs: $\eta_j^B$ | Level 2 |
| $\Gamma^W$ | $m_1 \times n_1$ | Within-level relationships from exog. to endog. LVs: $\xi_{ij}^W \rightarrow \eta_{ij}^W$ | Level 1 |
| $\Gamma^B$ | $m_2 \times n_2$ | Between-level relationships from exog. to endog. LVs: $\xi_j^B \rightarrow \eta_j^B$ | Level 2 |
| | | **Measurement Error Covariance Matrices** | |
| $\Theta_\varepsilon^W$ | $k \times k$ | Within-level endog. measurement error covariance: $\varepsilon_{ij}^W$ | Level 1 |
| $\Theta_\varepsilon^B$ | $k \times k$ | Between-level endog. measurement error covariance: $\varepsilon_j^B$ | Level 2 |
| $\Theta_\delta^W$ | $p \times p$ | Within level exog. measurement error covariance: $\delta_{ij}^W$ | Level 1 |
| $\Theta_\delta^B$ | $p \times p$ | Between-level exog. measurement error covariance: $\delta_j^B$ | Level 2 |
| | | **LV and Structural Residual Covariance Matrices** | |
| $\Psi^W$ | $m_1 \times m_1$ | Within-level structural residual covariance: $\zeta_{ij}^W$ | Level 1 |
| $\Psi^B$ | $m_2 \times m_2$ | Between-level structural residual covariance: $\zeta_j^B$ | Level 2 |
| $\Phi^W$ | $n_1 \times n_1$ | Within-level exog. LV covariance: $\xi_{ij}^W$ | Level 1 |
| $\Phi^B$ | $n_2 \times n_2$ | Between-level exog. LV covariance: $\xi_j^B$ | Level 2 |

Table 2: Overview of model parameters for the two-level random-intercept MLSEM model. The table displays the symbol used, the dimensions, the definitions and levels for each of the model parameters.

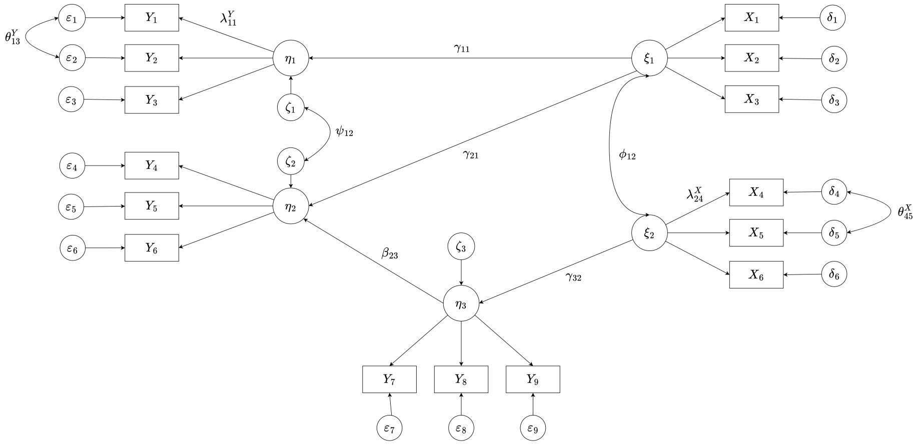


Figure 1: Path Diagram representation of a Single-Level SEM model. The SEM model contains three endogenous latent variables measured by three observed variables each and two exogenous latent variables also measured by three observed variables each. The model contains structural relationships from exogenous latent variables to endogenous latent variables, as well as relationships from an endogenous latent variable to another endogenous latent variable. The path diagrams also specifies (residual) correlations for the latent variables and measurement errors.

**Example of a Single-Level SEM Model: Path Diagram** MLSEM models, as well as SEM models, are often represented and specified with a graph. Rather than using matrix notation, the graph notation used directed edges (arrows), double headed arrows, rectangles and circles to specify relationships and variables in the model. Figure 1 provides an example of a path diagram for a single level model. Figure 2 contains an overview of the various elements that make up a path diagram along with their definition.

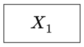
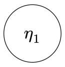
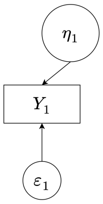
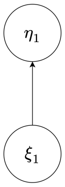
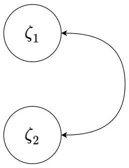
| Object | Definition |
|---|---|
|  | Rectangular or square box denotes an observed variable |
|  | Circles denote unobserved or latent variables, including measurement error |
|  | Arrows from latent to observed variables denote a factor loading. Arrows from a residual to observed variables denote measurement error |
|  | Directed arrows from a latent variable to another denote a structural relationship |
|  | Double headed arrows denote a residual correlation, i.e. not explained by structural model or measurement model |

Figure 2: This table contains the primary elements of a path diagram to specify SEM models. Next to each element, a brief definition of the element is provided.

The model specified in 1 has nine observed endogenous variables, $Y_i$, and six observed exogenous variables, $X_i$, with corresponding measurement errors, $\varepsilon_i$ and $\delta_i$:

$$Y_i = \begin{pmatrix} Y_i^1 \\ \vdots \\ Y_i^9 \end{pmatrix} , \quad X_i = \begin{pmatrix} X_i^1 \\ \vdots \\ X_i^6 \end{pmatrix} ,$$

$$\varepsilon_i = \begin{pmatrix} \varepsilon_i^1 \\ \vdots \\ \varepsilon_i^9 \end{pmatrix} , \quad \delta_i = \begin{pmatrix} \delta_i^1 \\ \vdots \\ \delta_i^6 \end{pmatrix} .$$

In addition, it contains three latent endogenous variables, $\eta_i$ , three structural residual terms, $\zeta_i$ and two latent exogenous variables, $\xi_i$ :

$$\eta_i = \begin{pmatrix} \eta_i^1 \\ \eta_i^2 \\ \eta_i^3 \end{pmatrix} , \quad \zeta_i = \begin{pmatrix} \zeta_i^1 \\ \zeta_i^2 \\ \zeta_i^3 \end{pmatrix} , \quad \xi_i = \begin{pmatrix} \xi_i^1 \\ \xi_i^2 \end{pmatrix} .$$

Since we have nine observed endogenous variables, $Y_i$ , and three latent endogenous variables, $\eta_i$ , we know that the endogenous factor loadings, $\Lambda_Y$ , should be a $(9 \times 3)$-matrix. If there is an arrow from the $n^{\text{th}}$ endogenous latent variable, $\eta_i^n$ , to the $m^{\text{th}}$ observed endogenous variable, $Y_i^m$ , we let $(\Lambda_Y)_{mn} = \lambda_{mn}^Y$ , otherwise $(\Lambda_Y)_{mn} = 0$ , i.e. its fixed to zero. The same logic applies to the exogenous factor loadings.

This yields the following factor loading matrices:

$$\Lambda_Y = \begin{pmatrix} \lambda_{11}^Y & 0 & 0 \\ \lambda_{21}^Y & 0 & 0 \\ \lambda_{31}^Y & 0 & 0 \\ 0 & \lambda_{42}^Y & 0 \\ 0 & \lambda_{52}^Y & 0 \\ 0 & \lambda_{62}^Y & 0 \\ 0 & 0 & \lambda_{73}^Y \\ 0 & 0 & \lambda_{83}^Y \\ 0 & 0 & \lambda_{93}^Y \end{pmatrix} , \quad \Lambda_X = \begin{pmatrix} \lambda_{11}^X & 0 \\ \lambda_{21}^X & 0 \\ \lambda_{31}^X & 0 \\ 0 & \lambda_{42}^X \\ 0 & \lambda_{52}^X \\ 0 & \lambda_{62}^X \end{pmatrix} .$$

**Remarks:**

- Setting, for example, $(\Lambda_Y)_{mn} = \lambda^Y_{mn}$ means that this coefficient has to be estimated and thus becomes a parameter.
- Its conventional to fix one factor loading per latent variable to be equal to 1, e.g. for $\Lambda_X$ we could let $\lambda^X_{11} = \lambda^X_{42} = 1$. This provides scale to the model and is necessary for model identification (see section 2.4 for details on identification). This is left out of this example to emphasize the link between the path diagram and the model parameters.

We have three arrows (e.g. $\gamma_{11}$) pointing from exogenous latent variables to endogenous latent variables and one arrow ($\beta_{23}$) from an endogenous latent variable to another endogenous variable, where entries that don not correspond to an arrow are again fixed to 0, as with the factor loadings. Since we have three endogenous latent variables and two exogenous latent variables, $B$ must be a $(3 \times 3)$-matrix and $\Gamma$ a $(3 \times 2)$-matrix.

This yields the following structural coefficient matrices:

$$B = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \beta_{23} \\ 0 & 0 & 0 \end{pmatrix} \quad , \quad \Gamma = \begin{pmatrix} \gamma_{11} & 0 \\ \gamma_{12} & 0 \\ 0 & \gamma_{23} \end{pmatrix} \quad ,$$

where again any entry of these matrices that does not correspond to an arrow is fixed to 0 .

Lastly, we specify the covariance matrices. Covariance matrices for the measurement errors and structural residuals are conventionally assumed to be diagonal, however, the path diagram specifies the presence of residual correlations. The covariance matrices need to be square matrices, so from the dimensions of their respective variables we derive that $\Theta_\varepsilon$ must be a $(9 \times 9)$-matrix, $\Theta_\delta$ a $(6 \times 6)$-matrix and $\Psi$ a $(3 \times 3)$-matrix.

This yield the following covariance matrices:

$$\Theta_\varepsilon = \begin{pmatrix} \theta^\varepsilon_{11} & \theta^\varepsilon_{12} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \theta^\varepsilon_{12} & \theta^\varepsilon_{22} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \theta^\varepsilon_{33} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \theta^\varepsilon_{44} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \theta^\varepsilon_{55} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \theta^\varepsilon_{66} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \theta^\varepsilon_{77} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \theta^\varepsilon_{88} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \theta^\varepsilon_{99} \end{pmatrix} ,$$

$$\Theta_\delta = \begin{pmatrix} \theta^\delta_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & \theta^\delta_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & \theta^\delta_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & \theta^\delta_{44} & \theta^\delta_{45} & 0 \\ 0 & 0 & 0 & \theta^\delta_{54} & \theta^\delta_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & \theta^\delta_{66} \end{pmatrix} ,$$

$$\Psi = \begin{pmatrix} \psi_{11} & \psi_{12} & 0 \\ \psi_{21} & \psi_{22} & 0 \\ 0 & 0 & \psi_{33} \end{pmatrix} ,$$

$$\Phi = \begin{pmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{pmatrix} ,$$

where the diagonal elements of these matrices (e.g. $\psi_{11}$) denote variances and the off-diagonal elements denote covariances (e.g. $\theta^\varepsilon_{12}$ ). As is required of covariance matrices, these matrices are symmetric, thus it must hold, for example, that$\psi_{21} = \psi_{12}$ .

**Remark:**

- Note that $\Phi$ is an exception to the previously stated rule that any unspecified covariance (or regression coefficient) is fixed to equal 0. No assumptions are made with regards to the covariance matrix of $\xi_i$ and thus we include all entries of the covariance matrix, specifically they are not constrained by model equations.

## 2.3 Likelihood and Covariance

The likelihood function in MLSEM represents the likelihood of observing the data given the model parameters. It forms the basis for estimation in both frequentist and Bayesian frameworks. For a two-level MLSEM model, the likelihood is derived from the model implied covariance matrix and the observed (sample) covariance matrix.

Using the model equations and model parameters, we can derive an expression of the covariance matrices (for both the within and between components) as a function of model parameters. This is often denoted in the following way:

$$\Sigma_Y^W = \Sigma_Y^W(\theta)\,, \qquad \Sigma_Y^B = \Sigma_Y^B(\theta)\,, \tag{2.3.1}$$

$$\Sigma_X^W = \Sigma_X^W(\theta)\,, \qquad \Sigma_X^B = \Sigma_X^B(\theta)\,, \tag{2.3.2}$$

$$\Sigma_{YX}^W = \Sigma_{YX}^W(\theta)\,, \qquad \Sigma_{YX}^B = \Sigma_{YX}^B(\theta)\,. \tag{2.3.3}$$

The complete expression of the likelihood function for two-level MLSEM, as implemented in Blavaan, can be found in (Rosseel, 2021)[17].

## 2.4 Specification and Identification

Specification in MLSEM refers to translating theory into a formal MLSEM framework, including paths between latent and observed variables as well as regression paths between latent variables in the structural model. Specification of a MLSEM can be done by constraining some parameters - usually equal to 0 - and letting other parameters vary freely, which will then be estimated. Specification often also includes conditions on covariances between residuals, which are often assumed to be independent. In two-level MLSEM, both the within-group (level 1) and between-group (level 2) structural and measurement models need to be specified.

A model is said to be identified if there is a unique set of parameters that can reproduce the model-implied covariance matrices. An alternative definition is that if a parameter can be written as a function of the observed covariance, then that parameter is identified; if all parameters are identified the model is identified (Bollen, 1989)[2]. Typical strategies to identify (ML-)SEM models is to fix one factor loading per latent variable to 1, to provide scale to the model or alternatively to fix the variance of latent variables to 1. Overparameterization can cause parts of the model to be under-identified. There are various heuristics to test wether a MLSEM model is identified, most notably the t-rule, which tests wether the amount of known information exceed the amount of unknown information (Brouwer, 2021)[3]. These heuristics, however, don't guarantee that the model is identified, as such it is a necessary but not sufficient condition. Ensuring model identification is important to get reliable and interpretable results. In Bayesian estimation, non-identified parameters can be overly influenced by priors.

# 3 Bayesian Estimation in MLSEM

In the previous chapter, we laid out the basics of MLSEM models, including model notation and derivation of the model implied covariance matrix. Frequentist estimation of MLSEM models is most often done with maximum likelihood estimation (ML), employing an iterative algorithm to obtain estimates. ML estimation often encounters problems with estimation of MLSEM models, especially in a small sample size context or when estimating many group-level variance components that are near or equal to zero.

Bayesian estimation is particularly suited for estimation in MLSEM, for example, in cases when sample sizes are small, asymptotical analysis is not reliable and can give too small confidence intervals and p-values (Hox, 2010)[11]. Bayesian analysis quantifies the uncertainty in the resulting estimates through the prior distribution, giving a probabilistic interpretation of the estimates that is valid even for small sample sizes. The use of prior distributions offers a way to incorporate prior knowledge into the estimation procedure, which has been shown to reduce mean square error (MSE) of slope estimates in MLSEM (Zitzmann et al., 2020) [1]. Another benefit of Bayesian estimation in MLSEM is the ability to take theoretical constraints into account by choice of priors or technical constraints regarding admissible parameter values. With the advent of computational methods such as Markov Chain Monte Carlo simulation (MCMC), Bayesian analysis and estimation of complex models, such as MLSEMs, has become easier and wide spread in statistical software (e.g., Blavaan in R), allowing estimation of models that were intractable with classic ML estimation (Depaoli, 2021)[7]. MCMC methods sample the posterior distribution using sampling algorithms offering additional benefits, such as the ability to directly transform samples to obtain new quantities of interest (e.g., direct- and total effects) and to incorporate the sampling of latent variables, offering an alternative to high dimensional numerical integration often encountered in ML estimation of multilevel latent variable models.

In this section we will discuss the use of Bayesian statistics in the context of MLSEM models. We will discuss the foundation of Bayesian inference, Bayes' Theorem, and cover the influence of priors on estimation of MLSEM models. The choice of prior is regarded as a subjective choice and has a great influence on posterior estimates. In small sample sizes, informative prior distributions can improve estimate accuracy and decrease variability (Depaoli, 2021)[7]. But even when no prior information is known, a weakly informative prior can be specified to regularize estimates (Depaoli, 2021)[7].

**Bayesian vs Frequentist** Bayesian statistics is a branch of statistics that makes use of the Bayesian interpretation of probability, which is that a probability represents a (subjective) degree of belief or certainty in an event, given our current state of knowledge. It's different from frequentist (classical) statistics in this regard, which interprets a probability as a long run relative frequency of an event in repeated identical experiments or draws.

Bayesian- and frequentist statistics differ most notably in the way they treat population parameters. In frequentist statistics, population parameters are interpreted to be fixed but unknown values, where as, Bayesian statistics treats population parameters as random variables with a distribution that represents our current state of knowledge or certainty about those parameters.
Frequentist statistics makes use of sampling distributions to quantify certainty of our estimates and hypothesis testing, where hypotheses may be accepted or rejected based on the likelihood of the observed data under that hypothesis. It can not, however, tell us how likely it is that a certain hypothesis is true; frequentist statistics fundamentally assumes that this is a fixed but unknown state of the world, and hence it can not have a probability assigned to it. Bayesian statistics, instead, relies on prior distributions that quantify our beliefs about the hypotheses and combines prior beliefs and observed data to update that certainty.
Bayesian statistics can be used to incorporate prior knowledge about the data, which can be especially helpful when little data is available. The results we get from Bayesian statistics are, however, very sensitive to the choice of priors, which is also subjective, where as frequentist methods only rely on data.

## 3.1 Bayes' Theorem

One of the most important concepts in Bayesian statistics and probability in general is Bayes' Theorem. It defines the conditional probability of an event conditional on another event.

**Theorem 3.1** (Bayes' Theorem)**.** *Let $(\Omega, \Sigma, \mathbb{P})$ be a probability space. Let $A, B \in \Sigma$ such that $\mathbb{P}(B) > 0$. Then we have that:*

$$\mathbb{P}(A|B) = \frac{\mathbb{P}(A \cap B)}{\mathbb{P}(B)} = \frac{\mathbb{P}(B|A)\mathbb{P}(A)}{\mathbb{P}(B)} \propto \mathbb{P}(B|A)\mathbb{P}(A) \tag{3.1.1}$$

We can also formulate Bayes' Theorem for the density of continuous random variables.

**Theorem 3.2** (Bayes' Theorem Continuous Form)**.** *Let $X$ and $\Theta$ be continuous random variables. Suppose the conditional density of $X$ is given by $p(x|\theta)$ and the marginal distribution of $\Theta$ is given by $\pi(\theta)$. Then we have that:*

$$\begin{aligned} p(\theta|x) &= \frac{p(x,\theta)}{p(x)} = \frac{p(x|\theta)\pi(\theta)}{p(x)} \\ &= \frac{p(x|\theta)\pi(\theta)}{\int p(x|\theta)\pi(\theta)d\theta} \propto p(x|\theta)\pi(\theta) \end{aligned} \tag{3.1.2}$$

### 3.1.1 Likelihood function, Prior & Posterior

Often, statistical models are defined using a distribution of the data conditional on the model parameters, denoted $p(x|\theta)$. When the data are observed and this

function is seen as a function of model parameters, we call it a likelihood function and denote it by $L(\theta|x)$.(Levy & Mislevy, 2016)[14]. In frequentist statistics the data are treated as random, modeled by this conditional distribution and the model parameters are treated as fixed. In Bayesian statistics we treat both the data and the parameters as random.
This furthermore requires us to specify a distribution on the parameters, $p(\theta)$, which is known as the prior distribution. Multiplying the likelihood function and the prior distribution on the parameters, we get the joint distribution on both the data and the parameters. Using Bayes' Theorem (theorem 3.2) we can then also calculate the distribution of the parameters conditional on the data, $p(\theta|x)$, which is known as the posterior distribution.

**Definition 3.1.** *Let $X$ be a random variable with density (or probability mass function for discrete $X$) $p_\theta$ depending on parameters $\theta$. Then, for fixed $x$, the likelihood function is the function:*

$$\theta \mapsto L(\theta; x) = p_\theta(x) \tag{3.1.3}$$

**Definition 3.2.** *Let $X$ be a random variable with likelihood function L($\theta$; x). A prior distribution is a probability distribution $\pi(\theta)$ on the parameters $\theta$.*

**Definition 3.3.** *Let $X$ be a random variable with likelihood function L($\theta$; x) and $\pi(\theta)$ a prior distribution on the parameter $\theta$.
The posterior distribution p($\theta|x$) is the conditional distribution of the parameter $\theta$ after observing the data. The posterior distribution for a given prior and likelihood function can be found using Bayes' Theorem.*

**Definition 3.4.** *Let $X$ be a random variable with likelihood function L($\theta$; x). A conjugate prior distribution, $\pi(\theta)$, is a prior distribution that, when combined with the likelihood function, results in a posterior distribution $p(\theta|x)$ that is in the same family of distributions as $\pi(\theta)$.*

### 3.1.2 Example: Application of Bayes' Theorem

Suppose we have $n$ independent draws, $X = (X_1, X_2, \ldots, X_n)$, from a normal distribution with unknown mean $\mu$ and known variance, $\sigma^2$. That is:

$$X_i \sim N(\mu, \sigma^2)\,. \tag{3.1.4}$$

Note that equation 3.1.4 specifies our statistical model and also specifies a likelihood function for the data. For convenience, we assume that we know the variance and are only interested in $\mu$. We assume a normal prior on $\mu$,

$$\mu \sim N(\mu_0, \tau_0^2)\,. \tag{3.1.5}$$

We can rewrite equations 3.1.4 and 3.1.5 into a likelihood function and prior distribution:

$$\begin{aligned} P(X|\mu) &= \prod_{i=1}^{n} P(x_i|\mu) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i-\mu)^2}{2\sigma^2}\right) , \\ P(\mu) &= \frac{1}{\sqrt{2\pi\tau_0^2}} \exp\left(-\frac{(\mu-\mu_0)^2}{2\tau_0^2}\right) , \end{aligned} \tag{3.1.6}$$

where $P(X|\mu)$ is the likelihood function and $P(\mu)$ is the prior distribution on $\mu$. We can then apply Bayes' Theorem to the likelihood function and prior distribution to get the posterior distribution,

$$\begin{aligned} P(\mu|X) &= \frac{P(X|\mu)P(\mu)}{P(X)} , \\ &\propto P(X|\mu)P(\mu) , \\ &\propto \exp\left(-\frac{1}{2}\left[\frac{n}{\sigma^2}(\bar{x}-\mu)^2 + \frac{1}{\tau_0^2}(\mu-\mu_0)^2\right]\right) , \end{aligned} \tag{3.1.7}$$

where $\bar{x}$ is the sample mean of the observed data and $P(X)$ is the marginal distribution of the data after integrating out the parameter $\mu$. This is only a function of the data and is thus regarded as a normalizing constant. We have furthermore left out any terms that are not functions of $X$ or $\mu$ and collected them in the normalizing constant.

## 3.2 Influence of Priors

**Influence of the Prior Distribution** Where frequentist statistics only rely on data and a likelihood function to produce estimates, Bayesian statistics also makes use of priors. The choice of prior is subjective and the resulting posterior distribution is thus also subjective. Levy & Mislevy (2016) [14]provide a great example describing the influence of the prior distribution on the resulting posterior distribution.

**Example: Beta-Binomial Model** Let $Y$ be a Binomial$(J, \theta)$ distributed variable, where $J$ is a fixed integer. Then for observed $y$ we have the following likelihood function:

$$L(\theta|y, J) = p(y|\theta, J) = \binom{J}{y} \theta^y (1-\theta)^{J-y} \propto \theta^y (1-\theta)^{J-y} . \tag{3.2.1}$$

We specify a Beta$(\alpha, \beta)$ prior on $\theta$:

$$\pi(\theta) = p(\theta|\alpha, \beta) = \frac{\theta^{\alpha-1}(1-\theta)^{\beta-1}}{B(\alpha, \beta)} \propto \theta^{\alpha-1}(1-\theta)^{\beta-1} , \tag{3.2.2}$$

where $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ and $\Gamma(x)$ is the Gamma-function.
Then, by Bayes' Theorem, we have that:

$$\begin{aligned} p(\theta|y, J) &\propto L(\theta; y, J)\pi(\theta) && (3.2.3) \\ &\propto \theta^{y+\alpha-1}(1-\theta)^{J-y+\beta-1} \,. && (3.2.4) \end{aligned}$$

The form of the resulting posterior distribution can be recognized as a Beta distribution (Levy & Mislevy, 2016)[14], i.e.:

$$p(\theta|y, J) = Beta(\theta|y + \alpha, J - y + \beta) \,. \qquad (3.2.5)$$

Note that, by definition 3.4, the Beta distribution is a conjugate prior as the resulting posterior distribution is also a Beta distribution.

To visualize the influence of prior distributions on posterior distributions, we look at the example by Levy & Mislevy (2016)[14]. The authors use the Beta-Binomial model specified above with $y = 7$ and $J = 10$ and consider three different priors. The priors used, the likelihood for the given parameters and the resulting posterior distributions are visualized in figure 3.

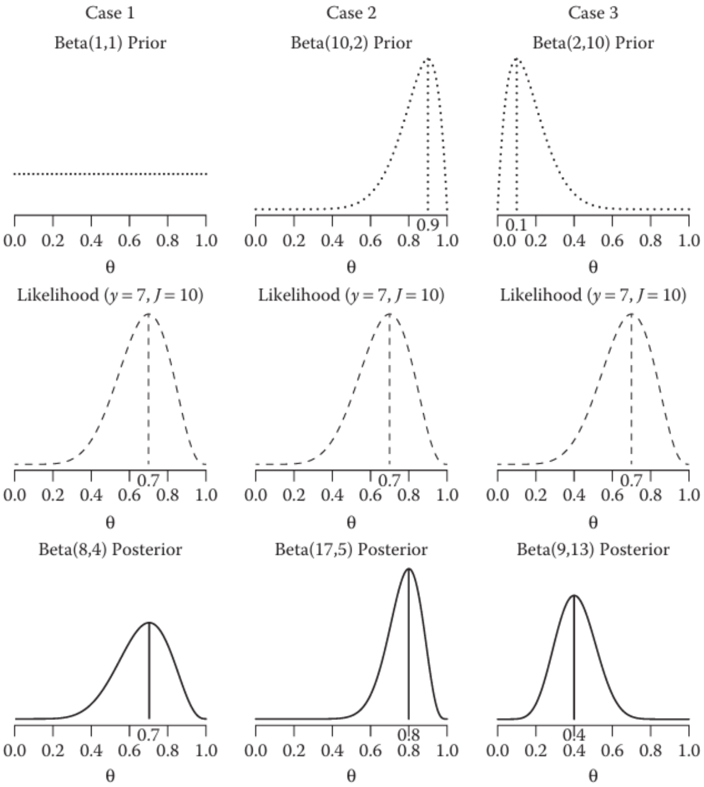


Figure 3: (Levy & Mislevy, 2016)[14]: Prior, likelihood function, and posterior for the Beta-binomial model with y = 7 with J = 10 and three different cases defined by different prior distributions. Reading down the columns presents the prior, likelihood, and posterior for each case. Where applicable, the mode of the distribution (solid) and the MLE (dotted) is indicated with a vertical line.

**Asymptotical Equivalence** It is important to note, however, that the Bayesian approach is asymptotically equivalent to the frequentist approach as the number of data points increases to infinity. In this case the posterior distribution limits to the likelihood function. The influence of the prior becomes less dominant in the posterior in this limit.

**Posterior Distribution** Using Bayesian stats we obtain a posterior distribution, which is the main result of the analysis (Levy & Mislevy, 2016)[14]. There are several ways we can summarize the posterior distribution. We could for example compute the posterior mean or posterior median, but we could also use the mode of this distribution, the value with the highest probability, this last estimate is known as the Maximum a Posteriori estimate (MAP).

Another common statistic is the credible interval, which is similar to a confidence interval from frequentist statistics. Where confidence are usually constructed using sampling distributions of the statistics, and thus relying on asymptotical properties and arguments, the credible interval is directly derived from the posterior distribution.
For example, a $100(1-\alpha)\%$ interval is constructed by computing the $100\frac{\alpha}{2}\%$ percentile and the $100(1-\frac{\alpha}{2})\%$ percentile. The former becomes the lower bound-and the latter becomes the upper bound of this interval.

**Model Comparison** Since the output of Bayesian estimators rely on posterior probability distributions, we can compute the actual probability of our hypothesis being true (assuming the likelihood and prior are correct). A comparable statistic from frequentist stats is the p-value, which is the probability of observing the data assuming the null hypothesis is true. The interpretation of p-values can be quite tricky and often misleading, where as the interpretation of the posterior probability is as an actual probability.

## 3.3 Bayesian Estimation of MLSEM models

Bayesian methods can help regularize estimates of parameters. This is especially useful for MLSEM models with many random effects variances, where ML estimates might be inadmissible, such as negative variances, or fail to converge. This is especially prevalent with small sample sizes and random effects variances equal to or near zero(Hox, J., 2012 & Depaoli, 2015)[11][6]. Variances (and covariances) are more difficult to estimate than means due to a flatter likelihood function (Depaoli, 2021)[7]. The use of priors can help to regularize estimates by shrinking them to the a-priori mean or by restricting inadmissible parameters by specifying a prior with support on allowed values only. When little prior knowledge is available a weakly informative prior can be specified over a range of 'reasonable' parameter values. Another benefit of Bayesian estimation in MLSEM models is the computational efficiency of sampling methods.

Rather than the iterative approach often applied by ML estimation, MCMC approximates the posterior distribution to obtain estimates. This may include latent variables used in the model, which prevents numerical integration required for some multilevel latent variable models. Multilevel latent variable models, such as MLSEMs, that include latent variable interactions are regarded as computationally infeasible for ML estimation (Muthen & Asparouhov, 2020)[15]. Bayesian estimation makes it possible to estimate more complex models, contributing to the types of research questions that can be answered with multilevel models.

**Prior Specification** To estimate MLSEM models with Bayesian estimation we need to specify priors for each parameter of the model. That is, we need to specify priors for the factor loadings ($\Lambda_X^W$, $\Lambda_X^B$, $\Lambda_Y^W$ and $\Lambda_Y^B$), regression coefficients ($B^W$, $B^B$, $\Gamma^W$ and $\Gamma^B$), measurement error (co)variances ($\Theta_X^W$, $\Theta_X^B$, $\Theta_Y^W$ and $\Theta_Y^B$), structural residual (co)variances ($\Psi^W$ and $\Psi^B$) and exogenous latent covariances ($\Phi^W$ and $\Phi^B$), at each level of the data, i.e. within- and between level for the two-level random intercept SEM model.

**Factor Loadings & Regression Coefficients** A common (weakly informative) prior for the factor loadings and regression coefficients is a (multivariate) normal distribution, where the normal prior for the factor loadings is centered at 1 and for the regression coefficients is centered at 0. For the factor loadings and regression coefficients an informative prior may also be specified with the normal distribution, where the variance of the prior is set to a lower value and the mean is set to an informative value. Zitzmann (2020) [1] has shown that the use of informative priors on regression coefficients can improve estimates for group level variances.

If we let $\theta_{normal} = (B, \Gamma, \Lambda)^T$ be a vector of parameters assumed to follow a multivariate normal distribution, we can specify the prior as follows:

$$\theta_{normal} \sim \mathcal{N}(\mu_{MVN}, \Sigma_{MVN}), \tag{3.3.1}$$

where $\mu_{MVN}$ is the mean hyperparameter in vector form and $\Sigma_{MVN}$ is the variance hyperparameter in covariance matrix form and $B = (B^W, B^B)$, $\Gamma = (\Gamma^W, \Gamma^B)$, $\Lambda = (\Lambda_Y^W, \Lambda_X^W, \Lambda_Y^B, \Lambda_X^B)$ are row-vectors containing the structural coefficients and factor loadings, respectively. Technically, this approach puts a normal prior on each of the individual elements of the matrices.

**Variance Parameters** A common prior for variance parameters is the inverse gamma distribution. For an unknown variance parameter $\sigma^2$ we can specify the inverse gamma prior as follows:

$$\sigma^2 \sim \mathcal{IG}[a_{\sigma^2}, b_{\sigma^2}], \tag{3.3.2}$$

where the hyperparameter $a$ and $b$ represent the shape and scale of the inverse gamma distribution, respectively. To obtain a proper prior, i.e. an actual probability distribution, both the hyper-parameters need to be set to positive values.

Equivalently, we could specify a gamma prior ($\mathcal{G}$) on the inverse of the variance, $1/\sigma^2$, also known as the precision, denoted as follows:

$$1/\sigma^2 \sim \mathcal{G}[a_{1/\sigma^2}, b_{1/\sigma^2}], \tag{3.3.3}$$

where the hyper-parameters $a$ and $b$ are again the shape and scale parameters, respectively and also need to have positive values to obtain a proper prior. Again, this prior specification is identical to specifying a inverse gamma prior on the variance.

We will also consider two alternatives to the inverse gamma prior ($\mathcal{IG}$). We consider a gamma prior, specified on the variance, $\sigma^2$, and on the standard deviation, $\sigma$. We denote this as follows:

$$\sigma^2 \sim \mathcal{G}[a_{\sigma^2}, b_{\sigma^2}]\,, \tag{3.3.4}$$

$$\sigma \sim \mathcal{G}[a_{\sigma}, b_{\sigma}]\,. \tag{3.3.5}$$

The reason we consider these alternatives is that these prior specifications, along with the gamma prior on the precision, are the available prior specifications of variance parameters in the R package Blavaan.

For an unknown variance parameter, we can specify the gamma prior on the variance directly, on the standard deviation or on precision, the latter two can be transformed back to variance. Because we specify the gamma prior on transformed variables, the effective density for the variance parameter changes. To highlight this effect, the three different options for prior (gamma distribution) specification on an unknown variance parameter are plotted in figure 4, with different parameters for the gamma distribution. For a full derivation of the densities, refer to the Appendix B. In the next chapter, we will study the influence of these different specifications on the estimation of variance parameters in a simulation study.

To highlight the differences in these priors, the different priors are plotted in figure 4.

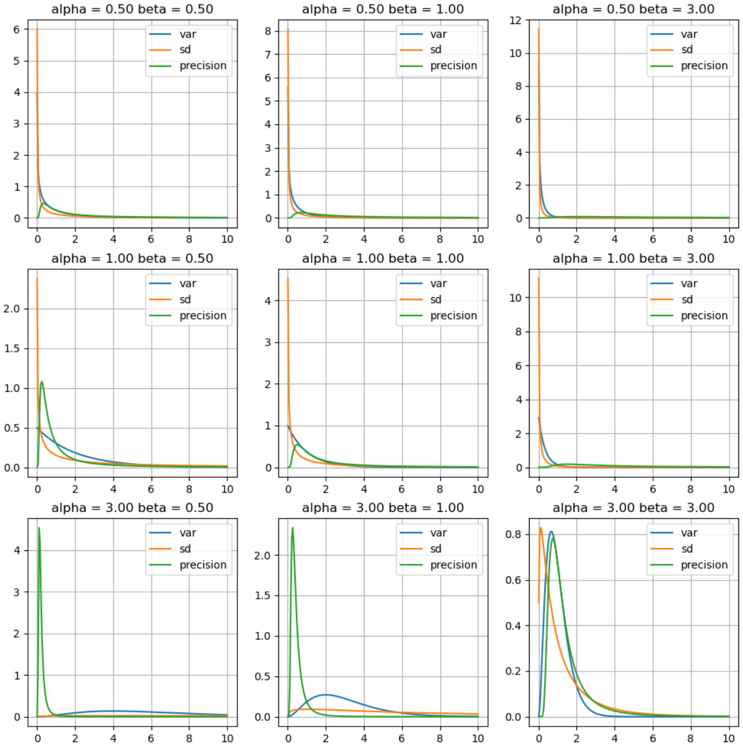


Figure 4: This figure shows multiple plots of the densities of three distributions of a variance parameter. The densities are annotated by 'var', 'sd' and 'precision', where each represents the density of a variance parameter when the gamma distributions is specified on the variance directly ('var'), on the standard deviation ('sd') or on the precision ('precision'). The plots in this figure are the three densities with varying parameters of the gamma distribution, as indicated by the title.

## 3.4 MCMC

For simple models with conjugate priors we have analytical solutions with the same form as the prior distribution. Many statistical models, however, do not enjoy the benefit of having conjugate priors. Computing the posterior distribution from the prior distribution and likelihood function can often prove to be intractable, as it often requires computing a high-dimensional integral. This integral is only a function of the observed data and is a constant. An alternative to this problem is to simulate values from the posterior distribution and use it as an empirical estimate. Markov Chain Monte Carlo (MCMC) refers to a collection of methods that construct a Markov chain which is sampled to obtain samples from a target distribution.

**Definition 3.5.** *A Markov chain is a sequence of random variables $(X_i)_{i>=0}$, satisfying the following property:*

$$\mathbb{P}(X_i|X_{i-1},...,X_0) = \mathbb{P}(X_i|X_{i-1}) \tag{3.4.1}$$

*Some additional properties of Markov chains:* (Levy & Mislevy, 2016)[14]

- *The Markov chain is called irreducible if from any point the chain can reach any other point with positive probability in a finite number of iterations.*
- *The Markov chain is called positive recurrent if in the long-run the chain visits each point an infinite number of times.*
- *The Markov chain is called aperiodic if it does not oscillate between two states with a regular period.*

If a Markov chain is irreducible, positive recurrent and aperiodic, it is called ergodic and will converge to a unique stationary distribution $\pi$.

**Markov Chain Monte Carlo** Markov Chain Monte Carlo (MCMC) refers to any method that samples data points from a distribution $\pi$ by constructing an ergodic Markov chain that converges to the target distribution $\pi$. In case of Bayesian statistics, we choose $\pi$ to be the posterior distribution of the parameters $\theta$. Several algorithms exist that implement this approach, each with different characteristics. (Metropolis-Hastings, Hamiltonian, Gibbs) MCMC methods are very popular because of the relative ease when dealing with high dimensional problems.

Instead of computing the posterior distribution and its moments analytically, we construct a Markov chain that converges to the posterior distribution and compute its moments or any other function of the samples empirically.
It is important to take enough samples from the posterior distribution as this method imposes its own sampling error on the estimates obtained, however this is under our control and can be chosen to satisfy our needs.

**Convergence** The convergence of MCMC methods refers to the point where the Markov chain reaches its target distribution. Convergence of MCMC methods is important because if a Markov chain converges, i.e. reaches its target distribution, the samples generated can be used as representative samples from our target distribution. If this fails however, these samples can not be used to reliably represent the posterior distribution.
MCMC usually provides a few key metrics to evaluate the validity of the posterior samples. MCMC methods usually run the sampling algorithms multiple times, each run is referred to as a chain, typically 4 chains are used.

If these chains are very similar, we assume that the algorithms converges. That is, the MCMC algorithm consistently provides a similar output and thus we conclude that the algorithm did not get stuck anywhere or was very sporadic in its runs. This is measured by a metric called $\hat{R}$ (”r-hat”), which is the ratio of the outputs of different chains. Values close to 1, $(0.995 \leq \hat{R} \leq 1.005)$, indicate that the independent chains converge to the same posterior distribution.

Another important metric to use is the effective sample size. The effective sample size is another way to quantify the certainty of our posterior distribution samples. High effective sample sizes indicate that we have a reliable estimate of our posterior distribution, while lower values indicate a low certainty. Effective sample size estimates the number of independent samples we got from the algorithm.

# 4 Simulation Study

In the previous chapter we discussed the basics of Bayesian inference as well as the influence of priors on estimation of parameters of the Beta-Binomial model. In this chapter, we will test three different priors for random effects variances. Insights in the influence of prior distributions can help guide us to pick the right prior for the application to real data. The results from this simulation study will be used in selecting a prior for random effects variances in the application to the PRIME research data.

We will specify a random intercept MLSEM model and simulate datasets based on various underlying parameter values. Specifically, we will vary the random effects variances and generate multiple datasets per model. These datasets will then be analyzed using the Blavaan R package, which was created for Bayesian estimation of (ML)SEM models. By applying different prior specifications on the random effects variances, we aim to compare and evaluate the performance of these priors. Through these simulations, we can better understand how different priors impact parameter recovery and estimation stability, particularly in multilevel contexts where data dependencies play a crucial role.

## 4.1 Simulation Study

We simulate a model with observed variables, $Y_{ij}^1$ , $Y_{ij}^2$ , $X_{ij}$ , that are transformed into latent variables for the estimation process, that is, we use latent variables that are measured by one observed variable with factor loading 1 and without measurement error. The reason we do this is because, as we will see in the next chapter, this is also done in the application of MLSEM to the PRIME research data. The model will also include a group level variable, $W_j$ . As was discussed in chapter MLSEM, to estimate MLSEM models an expression for the model-implied covariance matrices in terms of model parameters must be derived. Introducing an exogenous predictor variable at specific levels changes the expression of model-implied covariance at that level. This won't be discussed further in this thesis, see Bollen (1989)[2] for more details.

We introduce the following measurement equations. Note that each observed variable measures the corresponding latent variable perfectly. We do not specify a measurement equation for $W_j$ , since this is a variable that exists only at one level, in this case the group-level.

$$\begin{aligned} Y_{ij}^{W,1} &= \eta_{ij}^{W,1}\,, \\ Y_{ij}^{W,2} &= \eta_{ij}^{W,2}\,, \\ X_{ij}^{W} &= \xi_{ij}^{w} \end{aligned}$$

$$\begin{aligned} Y_{j}^{B,1} &= \eta_{j}^{W,1}\,, \\ Y_{j}^{B,2} &= \eta_{j}^{W,2}\,, \end{aligned}$$

The structural model is as follows. Note that we could have used $Y_{ij}$ and $X_{ij}$ directly in these equations, since they measure the latent variables perfectly. However, that is not done to keep this notation consistent with the previously introduced notation.

$$\begin{aligned} \eta_{ij}^{1} &= \beta_{12}^{W}\,\eta_{ij}^{2} + \gamma_{1}^{W}\,\xi_{ij} + \zeta_{ij}^{W,1} \\ \eta_{ij}^{W} &= \gamma_{2}^{W}\,\xi_{ij} + \zeta_{ij}^{W,2} \end{aligned}$$

$$\begin{aligned} \eta_{j}^{1} &= \mu_{Y}^{1} + \gamma_{1}^{B}\,W_{j} + \zeta_{j}^{B,1} \\ \eta_{j}^{2} &= \mu_{Y}^{2} + \gamma_{2}^{B}\,W_{j} + \zeta_{j}^{B,2}\,. \end{aligned}$$

Besides minor differences in this model from the model as introduced in chapter 2, its still possible to derive an expression for the model implied covariance in terms of the model parameters. This is also the reason that its possible to include arbitrary, even non-normal distributed, exogenous predictor variables. The covariance matrices of these exogenous predictors are estimated freely, i.e. equated to the sample covariance, and related to the covariance matrix of the latent endogenous variables through the regression equations that include them, similar to how the exogenous latent variables are related to the endogenous latent variables.

### 4.1.1 Simulation Design

We know from chapter 2 that we can express the covariance matrices as a function of model parameters, $\Sigma^{W}(\theta)\,,\Sigma^{B}(\theta)$. This allows us to sample data given the parameters of a model. In addition to model parameters, we also need to specify the number of groups, $J$, and the number of individuals within each group (group-size), $N_j$. For simplicity, we will be using the same group-size for each group, i.e. $N_j = N$. The focus of this simulation study is the effect of different priors on the estimation of random effects variances (variances of group-level components). The simulation will use three different priors for the variances to estimate the model. The estimates that this yields have some sampling variability and thus we will replicate the datasets and estimation procedure $M = 50$ times. Finally, since we don't apriori know the true value of the variance parameter in practical applications, we will also vary the variances in the model parameters. This gives us a sense of how well the different priors under study perform across values of the true parameter values.

### 4.1.2 Priors

We will be using the gamma distribution in this simulation study. Blavaan offers a few optional arguments to specify wether the gamma prior is specified on the standard deviation, precision or variance. Specifying the gamma prior on these options changes the prior slightly. See chapter 3 for more information on Bayesian estimation and the priors that will be used in the simulation study.

**Sampling Algorithm** Below we will reiterate some of the fixed settings of the simulation study:

- Number of groups, $J = 40$
- Group-size, $N = 5$
- Number of replications, $M = 50$

Given a parameter setting, $\theta$, number of groups, $J$ and cluster sizes, $N$, we can sample data using the following algorithm:

- For each $j = 1, \cdots, J$:
  - Sample $(Y_j^B, X_j^B) \sim \mathcal{N}(0, \Sigma^B(\theta))$
  - For each $i = 1, \cdots, N_j$:
    - Sample $(Y_{ij}^W, X_{ij}^W) \sim \mathcal{N}(0, \Sigma^W(\theta))$
    - Let $Y_{ij} = \mu_Y + Y_{ij}^W + Y_j^B$
    - Let $X_{ij} = \mu_X + X_{ij}^W + X_j^B$
- Store data in $D(\theta, J, N_j) = \{(Y_{ij}, X_{ij}) \,|\, j = 1, \cdots, J \quad \text{and} \quad i = 1, \cdots, N_j\}$

**Simulation Algorithm** For generality, the simulation algorithm is fully parameterized.

- $K(= 4)$ is number of parameter settings
- $L(= 3)$ is number of priors
- $\pi_n$ for $n = 1, \cdots, L$ different priors to test
- $\theta_i$ for $i = 1, \cdots, K$ parameter settings for which we can derive model-implied covariance $\Sigma_i = \Sigma(\theta_i)$

For $(\theta_m, \pi_n)$ with $n = 1, \cdots, L$ and $m = 1, \cdots, K$:

| Setting | Variance of $\zeta_j^{B,1}$ | Variance f $\zeta_j^{B,2}$ |
|---|---|---|
| 1 | 1.0 | 1.0 |
| 2 | 1.0 | 0.04 |
| 3 | 0.04 | 1.0 |
| 4 | 0.04 | 0.04 |

Table 3: Parameter settings that will be used to generate the data. We will only vary the variances of the random effects. All other parameters will be kept constant.

- For $p = 1, \cdots, M$:
  - Sample dataset $D_p(\theta_m)$
  - Estimate model with prior $\pi_n$ and store results in $R_p(\theta_m, \pi_n)$
- Return all results $R(\theta_m, \pi_n) = \{R_p(\theta_m, \pi_n) \quad | \quad p = 1, \cdots, P\}$

**Remarks** The results set was intentionally specified in an abstract manner to separate the specifics of the simulation study from the procedure of the simulation. In this simulation study, posterior means, convergence diagnostics (Rhat), and effective sample sizes (ESS) are recorded. Only results from converged runs are included in the final analysis. Convergence is determined by monitoring Rhat values and effective sample size (ESS), where Rhat values near 1 and sufficiently large ESS indicate that the Markov chains have reached a stable posterior distribution. Specifically, we will be filtering the estimation results with the following criteria ESS $\geq$ 100 and 0.95 $\leq$ Rhat $\leq$ 1.05. The default hyper-parameters for each prior are used to maintain consistency in scope.

**Parameter Settings** We design four main parameter settings to explore varying levels of random effects variance. This helps identify scenarios where certain priors might outperform others. For each setting, we compute both bias and RMSE across priors, noting patterns that may reveal when certain priors are more appropriate for use in multilevel models.

## 4.2 Blavaan

Blavaan is a R package that allows Bayesian estimation of MLSEMs. It uses a special type of model syntax (see table 4) to define the MLSEM model and then translates the specified model into JAGS and Stan code, which will be used to fit the models using Bayesian estimation.

Blavaan provides the option to specify priors that are included in JAGS or Stan to be used in model fitting. For multilevel SEM models, however, Blavaan only allows specific parametric priors to be specified on the parameters. For variance parameters, Blavaan offers a few optional arguments, it allows the user to specify wether to put the prior distribution on the standard deviation, the precision or on the variance itself.

| Operator | Pronunciation | Use |
|---|---|---|
| $\sim$ | Regressed on | Specify structural regression equations |
| $\sim=$ | Is measured by | Specify measurement equations |
| $\sim\sim$ | Variance | Denotes estimation of variance component |

Table 4: Operators used in the blavaan package to specify a SEM model.

**Syntax** Blavaan uses a text file or string to specify the models to estimate. In tabel 4, we will outline the operators used to specify a SEM model in Blavaan.

## 4.3 Results

In this section we will present plots that display the performance of the different priors used for the various parameter settings that are used in the simulation study. Performance will be measured by the estimator metrics bias and Root Mean Square Error (RMSE).

### 4.3.1 Estimators

We use various estimator metrics to assess estimation performance and parameter recovery. These estimator metrics are used to inform us on important qualitative properties of the estimators. The bias metric is intended to give an estimate of systemic difference between the parameter estimate and the true parameter. Root-mean-square-error (RMSE) is a common metric in statistics that combines both bias of the estimator and variance of the estimator. Variance of an estimator refers to the variability of the estimated value under a constant underlying true parameter. Estimators with high variance can be seen as less reliable, since their estimated values vary a lot and may be spread out over a large range, we can put less confidence in any single value that is produced by the estimator. RMSE combines measures of bias and variance into one single metric and is thus very useful to compare reliability of estimators.

Below we will give definitions for bias and RMSE.

**Definition 4.1.** *For an estimator $T$ and underlying parameter $\theta$ we define the bias of the estimator to be the following:*

$$Bias(T;\theta) = \mathbb{E}(T(X) - \theta) \tag{4.3.1}$$

Bias is the average difference between the estimated value of the parameter and the true value of the parameter.

**Definition 4.2.** *For an estimator $T$ and underlying parameter $\theta$, the root-mean-square-error or RMSE is defined as:*

$$RMSE(T;\theta) = \mathbb{E}((T(X) - \theta)^2) \tag{4.3.2}$$

RMSE combines a measure of bias and variance of the estimator. Variance of the estimator referes to the variability of the estimated value. Estimators with low RMSE are considered to be good estimators.

Four different plots, two plots containing the bias metric (figures 5 and 6) and two plots containing the RMSE metric (figures 7 and 8). For both bias and RMSE we have a single plot per variance parameter. Each of these plots is then divided into four different subplots, one for each parameter setting.

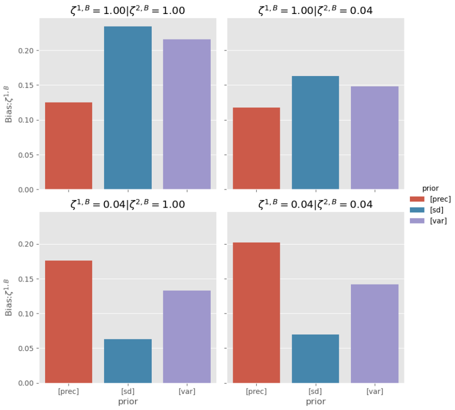


Figure 5: This figure shows the biases of the variance estimates of $\zeta_j^{B,1}$. The plot is faceted over the four different parameter settings used. The prior used for estimation is denoted by the color of the bar.

### 4.3.2 Conclusion

From the plots we can see that the effects of priors depend on the parameter settings. The prior standard deviation performs best when the true random effects variance is small. When the random effects variance is bigger, in the simulation this means a variance of 1, the precision prior performs best. Furthermore, from the bias plots it becomes apparent that all estimators overestimate the variance, indicated by a positive bias for all estimators. This result is inline with literature, where the overestimation of random effects variances is described as a common phenomenon. Much of the literature is devoted to priors on variances, when these variances are small. In multilevel modeling, researchers often include a variance / random component and use the estimate of this variance to decide whether it is large enough to be a significant contributor in the model. Because of this use case, especially the performance of priors in the small variance setting is interesting. Higher variances lead to bigger between-groups differences, since the parameters are significantly different. The standard deviation prior performs best in the low variance setting and is thus the most favorable option. It overestimates the variance the least of all available priors in this scenario and offers very comparable performance in the high variance setting. This leads us to conclude that it is the most robust prior of all the options.

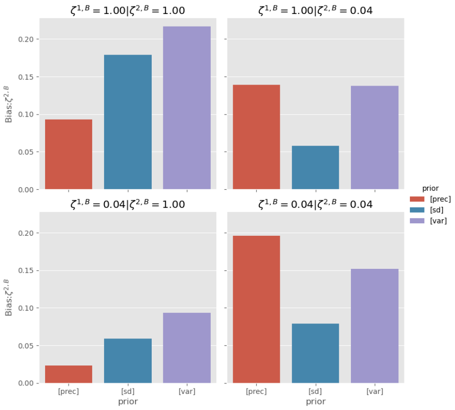


Figure 6: This figure shows the biases of the variance estimates of $\zeta_j^{B,2}$. The plot is faceted over the four different parameter settings used. The prior used for estimation is denoted by the color of the bar.

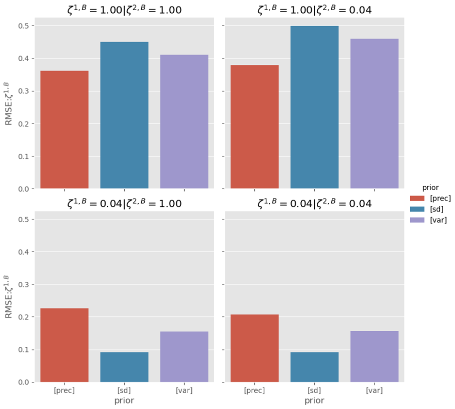


Figure 7: This figure shows the RMSE of the variance estimates of $\zeta_j^{B,1}$. The plot is faceted over the four different parameter settings used. The prior used for estimation is denoted by the color of the bar.

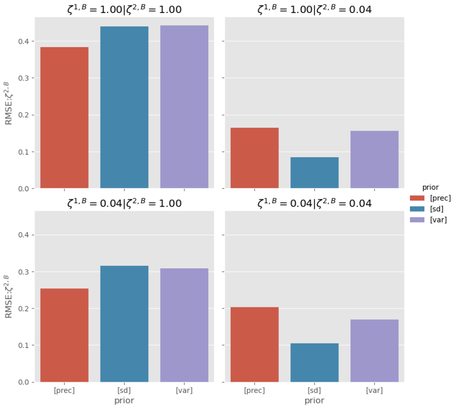


Figure 8: This figure shows the RMSE of the variance estimates of $\zeta_j^{B,1}$. The plot is faceted over the four different parameter settings used. The prior used for estimation is denoted by the color of the bar.

# 5 Application

## 5.1 Introduction

This chapter applies Bayesian MLSEM to PRIME research data gathered to examine how different cognitive load components interact with variables such as working memory capacity (WMC) and mental effort. By using the Blavaan package, we estimate the model and test hypotheses about these relationships. In the previous chapter, chapter 4, three different priors for variance parameters, specifically random effects variances, were investigated through a simulation study. From the literature its known that ML estimates of random effects variances that are close to zero, or otherwise close to a boundary, fail to converge or are underestimated. The simulation study showed that the gamma prior specified on the standard deviation, rather than on the variance directly, has the best performance, as measured by RMSE, for small true values of random effects variances. Therefore, this prior will be used in the estimation of the MLSEM model as applied to the PRIME research data.

## 5.2 PRIME Research

Self-monitoring and perceived mental effort (PME) play a crucial role in the process of regulating students' effort and pursuing their goals. However, how students experience and interpret their mental effort remains unexplored. PRIME research builds on the effort monitoring and regulation (EMR) framework proposed by de Bruin et al. (2020)[4], which integrates cognitive load theory (CLT) and self-regulated learning. The EMR framework proposes that effort monitoring occurs at two levels: an object level, where input is influenced by intrinsic-, extraneous- and germane cognitive load (ICL, ECL, GCL) and a meta-level, where meta-cognitive monitoring and regulation are influenced by the cues learners use and the beliefs they hold.
The PRIME research aims to explore how students perceive mental effort in a real-life setting by including various cognitive load measures, potential factors influencing their perception of mental effort and student performance.

### 5.2.1 SEM Model & Hypotheses

The aims of the PRIME research program can be summarized by the following list of effects and relationships:

- How do ICL, ECL and GCL relate to PME
- How do students' prior knowledge, self-efficacy and working memory capacity relate to PME
- How is the relationship between PME and students' prior knowledge, self-efficacy and working memory capacity mediated by ICL, GCL and ECL, respectively

To clarify these hypotheses in the context of the MLSEM framework, they will be be summarized in a table. The variable names will be related through arrows, the sign describing the hypothesis will be stated as well as the level of the relationship the hypothesis pertains to.

| **Relationship** | **Hypothesis** | **Level** |
|---|---|---|
| GCL → PME | + | Within |
| ECL → PME | + | Within |
| ICL → PME | + | Within |
| Prior Knowledge → PME | − | Between |
| Self-Efficacy → PME | Explorative | Within |
| WMC → PME | − | Between |
| Prior Knowledge → ICL → PME | Explorative | Between |
| Self-Efficacy → GCL → PME | Explorative | Within |
| WMC → ECL → PME | Explorative | Between |

Table 5: This table summarizes the hypotheses of the PRIME research formulated in links between variables in the MLSEM model. The table contains three columns: Relationship, containing the relationship to be studied as indicated with arrows, Hypothesis, indicating the sign of the relationship under the hypothesis or explorative to indicate the sign is not yet specified but is a quantity of interest, Level, indicating the level of the relationship, i.e. the level of the regression paths under study in the MLSEM model.

**Path Diagram** We can describe these research questions with a MLSEM model. As explained in chapter 2, we can specify MLSEM models using a path diagram. The model describing the PRIME research hypotheses is specified in figure 9.

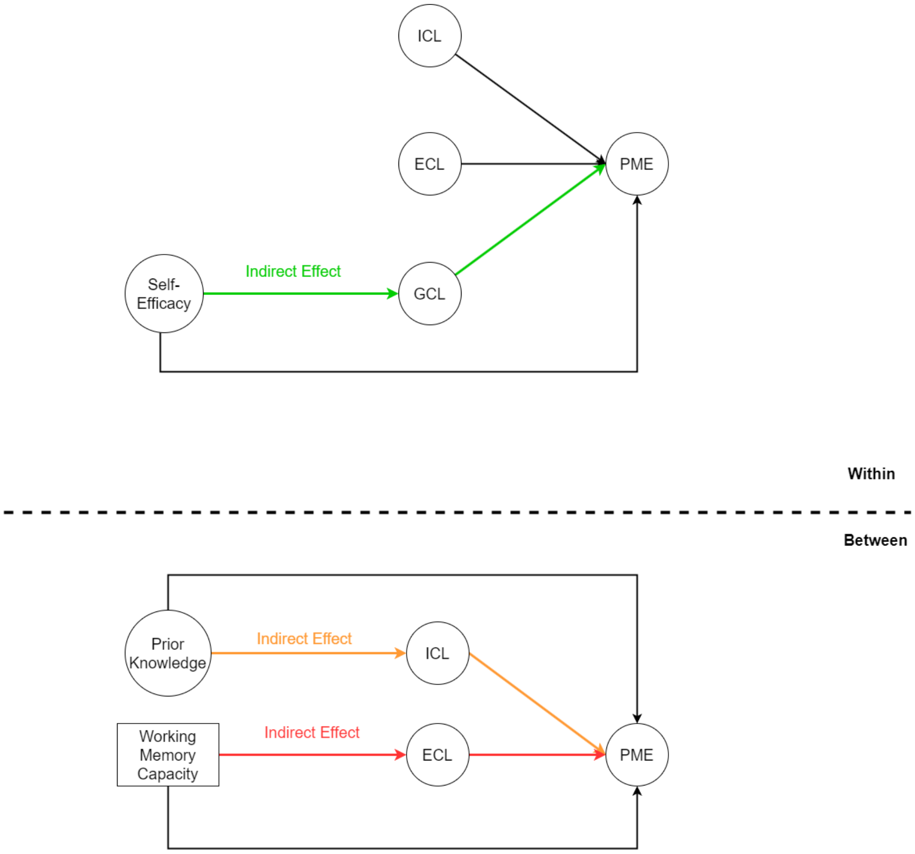


Figure 9: Path diagram specifying the MLSEM model that represent the PRIME research hypotheses. The figure includes specification of both the within and the between level. Some arrows have been colored to signify that they correspond to the same hypothesis on an indirect or mediated effect, as specified in table 5. Variables that exist at both the within and between level should be interpreted as their within and between components.

## 5.3 Data Collection

Data was collected through six surveys administered to students over a six-week period. After completing an online exercise on Grasple, students filled out surveys assessing cognitive load and related constructs. Each of the six surveys was identical, with the exception of the first and last surveys. The first survey included a single-item scale regarding prior knowledge in the content offered by the course and two cognitive tasks that measure the working memory capacity of the participants.
The survey responses are structured hierarchically, multiple students were asked on multiple occasions to fill out surveys. Surveys filled out by the same student share a common cause, namely the student that filled them out. The data is thus structured as surveys (within-level) nested within students (between-level).

### 5.3.1 Variables & Structure

In this section we will outline the variables used in the research and give a brief description of the variables. Furthermore, we will state and explain the level of the variable in terms of their collection method.

**Working Memory Capacity** Working memory capacity (WMC) refers to the ability of a person to work with- and hold multiple pieces of information in memory. If you are doing a task, the brain needs to process information related to the test. The amount of memory one can hold at any single time is often a constraining factor and is referred to as WMC.

Students take two cognitive tests that are meant to measure their working memory capacity (WMC). Their performance on these tests are then combined and averaged to get a score of their WMC. WMC is measured only once and not with every survey.

WMC is thought to be (approximately) constant within one individual. This makes WMC a group-level variable in our research.

**Measuring WMC** Working memory capacity of the participants was measured with a task proposed by Castro-Alonso et al. (2019) [5]. Each participant had to complete two sets of tasks. The maximum score a participant could obtain is 42 and the minimum score is 0.

**Perceived Mental Effort** Perceived mental effort is the level of cognitive strain a task causes within an individual. It is similar- and related to the difficulty of the task but there may be other factors that influence mental effort, such as various forms of cognitive load. Cognitive load causes a person to experience mental effort. This may be due to the difficulty of the task but can also be caused by distraction during the task. Mental effort is a subjective experience an individual has. Various forms of cognitive load cause the individual to experience mental effort. This research delves into the relationships between the various forms of cognitive load and the amount of mental effort students

experience.
On each survey students are asked to report the amount of mental effort they feel they invested in the exercise. Because mental effort is measured at each survey and can differ depending on the exercise, we include it as a within-level variable in our research.

**Measuring Mental Effort** Mental effort was measured with a single-item nine-point scale. The scale item used was proposed by Paas et al. (1992). The item read as follows:

*How much mental effort did you invest into this Grasple exercise?*

Respondents had to reply according to an ordinal scale, taking the following values in order:

- Very, very low mental effort
- Very low mental effort
- Low mental effort
- Rather low mental effort
- Neither low nor high mental effort
- Rather high mental effort
- High mental effort
- Very high mental effort
- Very, very high mental effort

**Difficulty** The difficulty of an exercise relates to the self-reported level of difficulty students experienced during the exercise. Difficulty is a subjective measure and may depend on the inherent difficulty of the topic but also on the prior knowledge of the student.
On each survey students are asked to report the difficulty of the exercises. Difficulty depends on the specific task and is thus measured after each task. This makes difficulty a within-level variable in our research.

**Measuring Difficulty** Difficulty was measured by a single-item nine-point scale, similar to mental effort. The item read as follows:

*Please rate the level of difficulty of this Grasple exercise for you.*

Respondents had to reply according to an ordinal scale, taking the following values in order:

- Very, very low difficulty

- Very low difficulty
- Low difficulty
- Rather low difficulty
- Neither low nor high difficulty
- Rather high difficulty
- High difficulty
- Very high difficulty
- Very, very high difficulty

**Prior Knowledge** The prior knowledge of students is a self-reported item students have to fill out. It asks students about their prior knowledge about- or experience with the general topics within this mathematics course, such as linear algebra and calculus.
Prior knowledge affects how students approach tasks and how their brain organizes information related to tasks. Students' prior knowledge is assessed at the start of the course and is constant within each student. This makes prior knowledge a group-level variable in our research.

**Measuring Prior Knowledge** Prior knowledge was measured by a single-item five-point scale. Respondents were asked to rate their prior knowledge in the content offered by the course. The item on this scale read as follows:

*Please rate your prior knowledge about the content offered in this mathematics course.*

Respondents had to reply according to an ordinal scale, taking the following values:

- None at all
- A little
- A moderate amount
- A lot
- A great deal

**Self-Efficacy** Self-efficacy refers to an individual's belief in their ability to successfully perform and complete tasks or achieve specific goals. In the context of cognitive load theory and mental effort, self-efficacy plays an important role in how learners approach, persevere, and ultimately succeed when faced with complex tasks. Self-efficacy is measured after each exercise, since it's dependent on the task. This makes self-efficacy a within-level variable in our research.

**Measuring Self-Efficacy** Self efficacy was measured on a 100-point single-item scale. The item on this scale read as follows:

> *Please indicate your degree of confidence in your ability to be successful in studying the content offered in this course by recording a number from 0-100.*

**Cognitive Load** Cognitive load refers to the mental resources and energy used when performing a task. There are different types of cognitive load related to different sources of resource usage. Cognitive load influences the mental effort experienced by students. The total amount of mental resources is constrained and (roughly) measured by WMC. Some types of cognitive load are related to the actual learning process, where as others are related to external factors or the intrinsic difficulty of the information. The learning process is benefited by dedicating resources to the learning itself. Thus higher levels of Germane CL are related to better learning of the topic. Because the brain is limited and CLs compete for these resources, it's important to minimize resources spent on external factors and maximize resources spent on learning. We will outline the definitions of three types of cognitive load as defined by John Sweller below.

**Extraneous Cognitive Load** Extraneous cognitive load (ECL) refers to the additional mental effort imposed on learners due to inefficient or poorly designed instructional materials, methods, or environmental factors. Unlike intrinsic load, this type of load does not arise from the complexity of the content itself but rather from obstacles that distract learners from the primary learning goal. It can include confusing instructions, irrelevant information, poorly organized content, or distractions like background noise.

**Germane Cognitive Load** Germane cognitive load (GCL) involves the mental effort required for constructing and strengthening schemas, which are organized knowledge structures that help learners categorize and understand complex concepts. Unlike intrinsic and extraneous load, germane load is considered beneficial to learning because it supports deeper processing and long-term knowledge retention.

**Intrinsic Cognitive Load** Intrinsic cognitive load (ICL) refers to the mental effort required to process the complexity of a task or learning material that is directly tied to its inherent structure and interrelated concepts. This type of cognitive load is determined by how much information a learner must simultaneously hold and manipulate in their working memory to fully understand the material. The complexity depends on factors like the number of elements, their interconnections, and the learner's prior knowledge.

**Measuring Cognitive Load** On each survey, students are asked to fill out items related to the three forms of cognitive load. These survey items

attempt to measure the degree of cognitive load experienced by the students during the exercise. This makes the three types of cognitive load within-level variables in our research.
Cognitive load is measured by a thirteen-item eleven-point Likert-type scale, four items for ICL, four items for ECL and five items for GCL. The scale is based on the ten-item scale proposed by Leppink et al.(2013)[12], with an additional mental-effort item for each of the three types of cognitive load (Leppink et al., 2014)[13], which increases reliability for ICL and ECL. The thirteen scale items are listed in table 6 along with the type of cognitive load they correspond to. Each of these items is responded to on an eleven-point Likert-type scale, ranging from 0 - "Not at all the case" to 10 - "Completely the case".

These variables provide a multi-faceted view of cognitive processing and effort across different tasks and individuals. The preliminary results of the PRIME study highlight that both within-level measures, such as PME and cognitive load components (ICL, ECL, GCL), and between-level measures, like WMC and prior knowledge, significantly affect student performance and perception of task difficulty. The table below summarizes the variables and their levels:

| **Cognitive Load Type** | **Survey Item** |
|---|---|
| ICL | The topics in this exercise were very complex. |
| ICL | The formulas used in this exercise were very complex. |
| ICL | In this exercise, very complex terms were mentioned. |
| ICL | I invested a very high mental effort in the complexity of this exercise. |
| ECL | The explanations and instructions in this exercise were very unclear. |
| ECL | The explanations and instructions in this exercise were full of unclear language. |
| ECL | The explanations and instructions in this exercise were, in terms of learning, very ineffective. |
| ECL | I invested a very high mental effort in unclear and ineffective explanations and instructions in this exercise. |
| GCL | This exercise really enhanced my understanding of the topics that were covered. |
| GCL | This exercise really enhanced my understanding of the formulas that were covered. |
| GCL | This exercise really enhanced my knowledge of the terms that were mentioned. |
| GCL | This exercise really enhanced my knowledge and understanding of how to deal with this kind of questions. |
| GCL | I invested a very high mental effort during this exercise in enhancing my knowledge and understanding. |

Table 6: This table shows the scale items of the thirteen-item scale used to measure cognitive load as experienced by the participants along with the type of cognitive load the item corresponds to (ICL, ECL, GCL).

| **Variable** | **Description** | **Level** |
|---|---|---|
| Perceived Mental Effort | Self-reported scale of cognitive resources allocated to the task | Within-level |
| Difficulty | Self-reported perception of difficulty | Within-level |
| WMC | Cognitive resource capacity | Group-level |
| Prior Knowledge | Self-reported understanding of course topics | Group-level |
| Self-Efficacy | Self-confidence in task performance | Within-level |
| ECL | Cognitive load related to external factors | Within-level |
| GCL | Cognitive load related to learning and building schemas | Within-level |
| ICL | Cognitive load related to the inherent complexity of the instructional material | Within-level |

Table 7: Overview of the variables used in the research. The table states the name of the variable, a short description, and the level of the variable in the research.

## 5.4 Data Preparation

In this section we will go over the data preparation process that is done before the data was used to fit the MLSEM model. Most of the data that we will be working with are scale items from a survey. The assumptions of the MLSEM modeling framework state that the observed variables must be (multivariate) normal distributed. The scale items used in the PRIME research are ordinal variables, i.e. they can be ordered from low to high. Ordinal variables are discrete variables and don't take values on a continuous spectrum, nor are they normal distributed. That means that these scale items can not be used directly in the MLSEM model. A common method to use ordinal items, like the scale items used in the PRIME research, in MLSEM is to model them with a latent threshold model. Unfortunately, this method is not available for multilevel SEM models in Blavaan, therefore the scale items need to be transformed before they are used in the MLSEM models. The transformations applied are intended to address the violations of the assumptions underlying the MLSEM model, specifically the assumption that the observed variables are normal distributed and thus continuous.
Furthermore, to get reliable estimates the group-sizes need to be of some minimum size. In this research the minimum group-size was taken to be three. The data will be filtered to satisfy this requirement, i.e. students (group-level) with less than three surveys (within-level) filled out will be excluded from the analysis.

### 5.4.1 Transforming Variables

To address the violations of the assumptions due to ordinal scale items, the data are transformed by the so called log-transform. Applying the log-transform can reduce skewness of the data and thus makes the data more symmetric (Fox, J., 2016) [9]. It also distributes the data over a larger range than the bounds imposed by the ordinal scale. Lastly, since the log-transform is a strictly increasing function, positive parameter estimates after applying the log-transform can still be interpreted as a positive effect. After applying the log-transform to the ordinal scale items, the scores of the multi-items scales are averaged, resulting in a sort of sum-score. This sum-score is used directly as a value of the latent variable corresponding to the survey items. Using sum-scores instead of relating observed variables to latent variables through the measurement model is allowed if Cronbachs' alpha is sufficiently large ($\alpha \geq 0.7$).

The log-transform is defined as follows, where $f_N$ denotes the log-transform and $N$ denotes the maximum value the ordinal variable can take:

$$p_N(x) = \frac{x + \frac{1}{2}}{N + 1}\,, \tag{5.4.1}$$

$$f_N(x) = \log\left(\frac{p}{1 - p}\right) \tag{5.4.2}$$

The log-transform is applied to the self-efficacy, difficulty, PME, ICL, ECL and GCL. The resulting transformed values for ICL, ECL and GCL are then averaged over the different scale items relating to that type of cognitive load. The code that applies this transformation can be found in appendix D.

### 5.4.2 Filtering Data

Before proceeding the analysis further, the data is filtered according to the following requirements:

- Filtered students based on number of surveys filled out; minimum 3 surveys.
- Filtered students based on within student variance, must have within level variance.

## 5.5 Model Estimation

Using Blavaan, we estimate the SEM model and assess model fit using Bayesian methods. The simulation study results inform our choice of priors, with the gamma distribution on the standard deviation (SD) of random effects proving robust across scenarios. Key findings include the influence of WMC on cognitive load components and the mediation effects of difficulty and mental effort.

While posterior predictive checks are not available in Blavaan, convergence diagnostics provide crucial insights into model stability. Metrics such as R-hat values and effective sample size (ESS) confirm that the Markov chains have converged to a stable posterior distribution. The simulation studies further support the reliability of the parameter estimates, demonstrating consistent results across multiple datasets and parameter settings.

## 5.6 Results

In this section, the hypothesis will be answered based on parameter estimates. The full resulting parameter estimates are listed in table 8. Indirect or mediated effects are the product of the relationships comprising them, e.g. the indirect relationship between prior knowledge and PME, mediated by ICL, is the relationship between PME and ICL multiplied by the relationship between ICL and PME, both at the between level in this case. If one of the constituent relationships is insignificant, the indirect relationship will be regarded as insignificant.

**Relationship between GCL and PME** The estimated relationship between GCL and PME is 0.040, which is a positive as hypothesized, however the 95% credible interval has a lower bound of −0.082, which indicates that the effect might not be positive. It also has an upped bound of 0.160, indicating that the posterior distribution is positively skewed, supporting the positive parameter estimate. Nonetheless, it will be regarded as an insignificant result.

**Relationship between ECL and PME** The estimated relationship between ECL and PME is −0.002, which is slightly negative contrary to the hypothesis. It has a credible interval ranging from −0.096 to 0.091, indicating that the estimated negative effect is insignificant.

**Relationship between ICL and PME** The estimated relationship between ICL and PME is 0.347, which is a positive effect as hypothesized. It has a credible interval ranging from 0.246 to 0.448 indicating that the relationship is significant and supports the hypothesis.

**Relationship between Prior Knowledge and PME** The estimated relationship between prior knowledge and PME is −0.041, which is a negative effect as hypothesized. It has a credible interval ranging from −1.400 to 1.125 indicating the estimated negative is insignificant.

**Relationship between Self-Efficacy and PME** The estimated relationship between self-efficacy and PME is 0.076, which is positive. There was not a hypothesized sign on this relationship. It has a credible interval ranging from −0.061 to 0.207, indicating that the positive effect was not significant.

**Relationship between WMC and PME** The estimated relationship between WMC and PME is $-0.001$, which is negative as hypothesized. It has a credible interval ranging from $-0.016$ to 0.014 indicating that the estimated negative effect is not significant.

**Indirect relationship between Prior Knowledge and PME** The estimated indirect effect between prior knowledge and PME, mediated by ICL, is 0.005. Since the relationship between prior knowledge and PME is insignificant, the estimated indirect effect is insignificant.

**Indirect relationship between Self-Efficacy and PME** The estimated indirect effect between self-efficacy and PME, mediated by GCL, is 0.003. Since both constituent relationships are insignificant, the estimated indirect effect is insignificant.

**Indirect relationship between WMC and PME** The estimated indirect effect between WMC and PME, mediated by ECL, is 0.002. Since both constituent relationships are insignificant, the estimated indirect effect is insignificant.

Table 8: Unstandardized Posterior Estimates with 95% Credible Intervals

| **Readable** | **Mean** | **SD** | **CI lower** | **CI upper** | **Type** | **Level** |
|---|---:|---|---:|---:|---|---|
| MentalEffort ~ GCL (within) | 0.040 | 0.062 | -0.082 | 0.160 | regression | within |
| MentalEffort ~ ECL (within) | -0.002 | 0.047 | -0.096 | 0.091 | regression | within |
| MentalEffort ~ SelfEfficacy (within) | -0.094 | 0.059 | -0.212 | 0.021 | regression | within |
| MentalEffort ~ ICL (within) | 0.347 | 0.052 | 0.246 | 0.448 | regression | within |
| GCL ~ SelfEfficacy (within) | 0.076 | 0.068 | -0.061 | 0.207 | regression | within |
| MentalEffort~~MentalEffort residual variance (within) | 0.171 | 0.019 | 0.138 | 0.211 | residual variance | within |
| GCL~~GCL residual variance (within) | 0.275 | 0.029 | 0.223 | 0.336 | residual variance | within |
| MentalEffort ~ PriorKnowledge (between) | -0.041 | 0.545 | -1.400 | 1.125 | regression | between |
| MentalEffort ~ SelfEfficacy (between) | -0.058 | 0.086 | -0.235 | 0.106 | regression | between |
| MentalEffort ~ Globalscore_WMC1 (between) | -0.001 | 0.008 | -0.016 | 0.014 | regression | between |

| Readable | Mean | SD | CI lower | CI upper | Type | Level |
|---|---:|---:|---:|---:|---|---|
| MentalEffort ˜ GCL (between) | 0.239 | 0.209 | -0.165 | 0.663 | regression | between |
| MentalEffort ˜ ECL (between) | 0.091 | 0.085 | -0.076 | 0.262 | regression | between |
| MentalEffort ˜ ICL (between) | 0.025 | 4.092 | -10.970 | 9.064 | regression | between |
| ICL ˜ PriorKnowledge (between) | -0.127 | 0.042 | -0.208 | -0.045 | regression | between |
| ECL ˜ Globalscore_WMC1 (between) | -0.012 | 0.018 | -0.049 | 0.024 | regression | between |
| MentalEffort˜˜MentalEffort residual variance (between) | 0.031 | 0.022 | 0.000 | 0.081 | residual variance | between |
| ICL˜˜ICL residual variance (between) | 0.021 | 0.028 | 0.000 | 0.097 | residual variance | between |
| ECL˜˜ECL residual variance (between) | 0.581 | 0.160 | 0.334 | 0.970 | residual variance | between |
| MentalEffort intercept (between) | 0.536 | 1.149 | -1.801 | 3.372 | intercept | between |
| ICL intercept (between) | 0.247 | 0.143 | -0.027 | 0.528 | intercept | between |
| ECL intercept (between) | -0.706 | 0.564 | -1.784 | 0.410 | intercept | between |

# 6 Conclusion

This thesis aimed to explore available prior distributions within 'Blavaan' for Multilevel Structural Equation Models (MLSEMs) and to provide guidelines for prior specification for random effects variances within 'Blavaan'. The literature indicates that Bayesian estimation of MLSEM is a promising field that enables more complex models, and thus more complex research questions, to be estimated. This thesis specifically investigates the prior specification for random effects variances because of two reasons: random effects variances are often unreliably estimated by maximum likelihood estimation and prior specification was shown to be influential on posterior estimates and random effects variances.

The research introduced the key concepts and mathematical foundations of Bayesian MLSEM. Chapter 2 focused on the foundation of MLSEM, covering model specification in terms of model equations, model assumptions and model parameters. This chapter introduced the foundations of estimation of MLSEM models by showing how the model-implied covariance is derived from model parameters, with the full derivation being given Appendix C. Chapter 3 delved into the basics of Bayesian inference, the influence of prior distributions and the extension of Bayesian estimation to MLSEM models. This chapter also introduced the priors that are used in the simulation study.

In chapter 4, a simulation study was conducted to examine the effect of prior specification on estimation accuracy. A significant finding from this study was that the gamma prior, when specified on the standard deviation rather than directly on the variance, demonstrated the best performance, as measured by Root Mean Squared Error (RMSE), particularly for small true values of random effects variances. This informed the choice of prior used in the application of the MLSEM model to real-world data from the PRIME research study.

The application chapter focused on analyzing the PRIME research data to understand how different cognitive load components interact with variables like working memory capacity (WMC) and mental effort. The Blavaan package was utilized for model estimation and hypothesis testing. The thesis contributes to the field by addressing the need for guidelines regarding prior specification in Bayesian MLSEM, emphasizing their influence on estimation accuracy and model stability. The insights gained, particularly regarding the effectiveness of the gamma prior on standard deviation, offer valuable guidance for researchers employing Bayesian MLSEM in diverse disciplines.

# 7 Discussion

This thesis investigated the role of prior specification for random effects variances in Bayesian Multilevel Structural Equation Models (MLSEM), with particular attention to the prior options available in the R package blavaan. Using both a simulation study and an applied example, we demonstrated how different prior specifications influence parameter recovery, especially in models with small group-level variances. While Blavaan provides an user-friendly interface for Bayesian estimation of MLSEM models, one key limitation was the limited set of priors that are available in Blavaan. The priors were constrained to the gamma-type priors, that is the gamma prior specified on the variance, the precision and the standard deviation. Many more priors have been suggested in the literature for the estimation of variance parameters, including the half-Cauchy, half-Students'-t, and half-normal distributions. This limited set of priors prevents researchers from tailoring distributions to their research.

While these priors are currently not available in Blavaan, future research into more prior specifications on variance parameters can prove both insightful as provide guidelines to researchers for prior specification in software packages such as Blavaan.

This thesis focused on random-intercept MLSEM models, which decompose the data into components between components and describe them separately with a MLSEM model. Although this model aligns with the research questions of the applied case (PRIME Research), not including random slopes and interaction effects limits the generalizability to more flexible MLSEM structures.

Taken together, these results emphasize the importance of thoughtful prior specification in Bayesian MLSEM, particularly when estimating random effects variances. While blavaan lowers the barrier to Bayesian SEM estimation, the current limitations in its prior specification framework call for further development. Expanding prior options and improving transparency in how priors are implemented could significantly enhance its utility for applied researchers.

# A Invertibility Condition

## A.1 Directed Acyclic Graph

Let $G = (V, E)$ be a directed graph. Then $G$ is a directed acyclic graph (DAG) if and only if its vertices can be put in topological ordering. That is, the vertices can be labeled $v_1, v_2, \ldots, v_n$ such that for every edge $(v_i, v_j) \in E$ we have that $ij$.

Let $A$ be the adjacency matrix of $G$ under this topological ordering defined as follows:

$$A_{ij} = \begin{cases} 1 \text{ if } (v_i, v_j) \in E \text{ else } 0 \end{cases} \tag{A.1.1}$$

Given the topological ordering, for any edge $(v_i, v_j) \in E$, we have that $ij$, thus we have that for any $i \geq j$ no such edge can exist and hence $A_{ij} = 0$. Thus $A$ is strictly upper-triangular.

Now let $G = (V, E)$ be a DAG on $n$ vertices. Let $A$ be its adjacency matrix under an arbitrary labeling and $\hat{A}$ be its adjacency matrix under the topological ordering, which is strictly upper triangular by the previous result. Let $P \in \{0, 1\}^{n \times n}$ be a permutation matrix representing the topological ordering.

## A.2 Nilpotency of Strictly Upper-Triangular Matrices

### Theorem

Let $A \in \mathbb{R}^{n \times n}$ be a strictly upper triangular matrix, i.e.,

$$A_{ij} = 0 \quad \text{for all } i \geq j.$$

Then $A$ is nilpotent. In fact,

$$A^n = 0.$$

### Proof (by induction on $n$)

**Base Case:** $n = 1$ A strictly upper triangular $1 \times 1$ matrix must be zero:

$$A = [0] \quad \Rightarrow \quad A^1 = 0.$$

So the base case holds.

**Inductive Hypothesis** Assume that the statement holds for all strictly upper triangular matrices of size $(n-1) \times (n-1)$. That is, if $B \in \mathbb{R}^{(n-1) \times (n-1)}$ is strictly upper triangular, then

$$B^{n-1} = 0.$$

**Inductive Step** Let $A \in \mathbb{R}^{n \times n}$ be a strictly upper triangular matrix. Write $A$ in block form:

$$A = \begin{bmatrix} 0 & u^\top \\ 0 & B \end{bmatrix},$$

where

- 0 in the top-left is a scalar (the $(1, 1)$ entry),
- $u^\top \in \mathbb{R}^{1 \times (n-1)}$,
- $0 \in \mathbb{R}^{(n-1) \times 1}$,
- $B \in \mathbb{R}^{(n-1) \times (n-1)}$ is strictly upper triangular.

We compute powers of $A$ by induction. Suppose that for some $k \geq 1$,

$$A^k = \begin{bmatrix} 0 & v_k^\top \\ 0 & B^k \end{bmatrix}.$$

Then,

$$A^{k+1} = A \cdot A^k = \begin{bmatrix} 0 & u^\top \\ 0 & B \end{bmatrix} \begin{bmatrix} 0 & v_k^\top \\ 0 & B^k \end{bmatrix} = \begin{bmatrix} 0 & u^\top B^k \\ 0 & B^{k+1} \end{bmatrix}.$$

Hence, we obtain the recurrence relation:

$$v_{k+1}^\top = u^\top B^k,$$

and in general:

$$v_k^\top = u^\top B^{k-1}, \quad \text{for } k \geq 1.$$

Since $B \in \mathbb{R}^{(n-1) \times (n-1)}$ is strictly upper triangular, by the inductive hypothesis we have:

$$B^{n-1} = 0 \quad \Rightarrow \quad v_n^\top = u^\top B^{n-1} = 0.$$

Therefore,

$$A^n = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = 0.$$

Thus, $A$ is nilpotent with index at most $n$.

## A.3 Result for nilpotent matrices

**Theorem** Let $B \in \mathbb{R}^{n \times n}$ be a nilpotent matrix of order $p \leq n$. Then $(I - B)$ is invertible and is given by $\sum_{i=0}^{\infty} B^i$

**Proof** Define $T : \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n}$ as follows:

$$T(B) := \sum_{i=0}^{\infty} B^i \tag{A.3.1}$$

Then if $B$ is nilpotent of index $p$, we have the following:

$$\begin{aligned}
T(B) &= \sum_{i=0}^{\infty} B^i && \text{(A.3.2)}\\
&= \sum_{i=0}^{p-1} B^i + \sum_{i=p}^{\infty} B^i && \text{(A.3.3)}\\
&= \sum_{i=0}^{p-1} B^i \,, && \text{(A.3.4)}
\end{aligned}$$

this follows from the fact that $B^p = 0$.
To show that $T(B) = (I-B)^{-1}$, we show that $(I-B)T(B) = I = T(B)(I-B)$.

$$\begin{aligned}
(I-B)T(B) &= T(B) - B\,T(B) && \text{(A.3.5)}\\
&= \sum_{i=0}^{p-1} B^i - \sum_{i=0}^{p-1} B\,B^i && \text{(A.3.6)}\\
&= \sum_{i=0}^{p-1} B^i - \sum_{i=0}^{p-1} B^{i+1} && \text{(A.3.7)}\\
&= \sum_{i=0}^{p-1} B^i - \sum_{i=1}^{p} B^i && \text{(A.3.8)}\\
&= \sum_{i=0}^{p-1} B^i - \sum_{i=1}^{p-1} B^i - B^p && \text{(A.3.9)}\\
&= \sum_{i=0}^{0} B^i - B^p && \text{(A.3.10)}\\
&= B^0 - B^p = I - 0 = I && \text{(A.3.11)}
\end{aligned}$$

Similarly, it follows that $T(B)(I-B) = I$. Hence $T(B)$ is the inverse of $(I-B)$.

# B Deriving the Density of the Variance $\sigma^2$

We consider two cases for deriving the density of the variance $\sigma^2$:

## Case 1: Standard Deviation $\sigma \sim$ Gamma$(\alpha, \beta)$

Let the standard deviation follow a Gamma distribution:

$$\sigma \sim \text{Gamma}(\alpha, \beta)$$

with probability density function:

$$f_\sigma(s) = \frac{\beta^\alpha}{\Gamma(\alpha)} s^{\alpha-1} e^{-\beta s}, \quad s > 0$$

We define:

$$v = \sigma^2 \quad \Rightarrow \quad \sigma = \sqrt{v}$$

Using the change-of-variable theorem, if $Y = g(X)$ is a one-to-one differentiable transformation of a random variable $X$, then (see [10]):

$$f_Y(y) = f_X(g^{-1}(y)) \left| \frac{d}{dy} g^{-1}(y) \right|$$

Here, $v = \sigma^2$, so:

$$\frac{d\sigma}{dv} = \frac{1}{2\sqrt{v}}$$

Thus, the density of $v$ becomes:

$$f_V(v) = f_\sigma(\sqrt{v}) \cdot \left| \frac{d}{dv} \sqrt{v} \right| = f_\sigma(\sqrt{v}) \cdot \frac{1}{2\sqrt{v}}$$

Substituting the expression for $f_\sigma$, we get:

$$f_V(v) = \frac{\beta^\alpha}{\Gamma(\alpha)} (\sqrt{v})^{\alpha-1} e^{-\beta\sqrt{v}} \cdot \frac{1}{2\sqrt{v}} = \frac{\beta^\alpha}{2\Gamma(\alpha)} v^{\frac{\alpha}{2}-1} e^{-\beta\sqrt{v}}, \quad v > 0$$

## Case 2: Precision $\tau = \frac{1}{\sigma^2} \sim$ Gamma$(\alpha, \beta)$

Assume the precision is Gamma distributed:

$$\tau = \frac{1}{\sigma^2} \sim \text{Gamma}(\alpha, \beta)$$

Let:

$$v = \sigma^2 = \frac{1}{\tau} \quad \Rightarrow \quad \tau = \frac{1}{v}, \quad \frac{d\tau}{dv} = -\frac{1}{v^2} \Rightarrow \left| \frac{d\tau}{dv} \right| = \frac{1}{v^2}$$

The PDF of $\tau$ is:

$$f_\tau(t) = \frac{\beta^\alpha}{\Gamma(\alpha)} t^{\alpha-1} e^{-\beta t}, \quad t > 0$$

Therefore, the density of $v$ is:

$$f_V(v) = f_\tau(1/v) \cdot \frac{1}{v^2} = \frac{\beta^\alpha}{\Gamma(\alpha)} \left(\frac{1}{v}\right)^{\alpha-1} e^{-\beta/v} \cdot \frac{1}{v^2}$$

Simplifying:

$$f_V(v) = \frac{\beta^\alpha}{\Gamma(\alpha)} v^{-\alpha-1} e^{-\beta/v}, \quad v > 0$$

This is the inverse gamma distribution:

$$v \sim \text{Inv-Gamma}(\alpha, \beta)$$

# C Derivation Model-Implied Covariance Matrix

For individual $i = 1, \cdots, N_j$ nested in cluster $j = 1, \cdots, J$ let:

- $Y_{ij}$ be a $k$-variate vector containing the endogenous observed variables
- $X_{ij}$ be a $p$-variate vector containing the exogenous observed variables
- For simplicity of notation, let $Z_{ij} = \begin{pmatrix} Y_{ij} \\ X_{ij} \end{pmatrix}$ be a $(k+p)$-variate vector containing all observed data

The data, $Y_{ij}$ and $X_{ij}$ is assumed to follow a joint normal distribution. That is:

$$Z_{ij} \sim \mathcal{N}(\mu_Z, \Sigma_Z)\,, \tag{C.0.1}$$

$$\mu_Z = \begin{pmatrix} \mu_Y \\ \mu_X \end{pmatrix}\,, \tag{C.0.2}$$

$$\Sigma_Z = \begin{pmatrix} \Sigma_Y & \Sigma_{YX} \\ \Sigma_{XY} & \Sigma_X \end{pmatrix}\,, \tag{C.0.3}$$

where $\Sigma_Y$ is the covariance matrix of $Y_{ij}$, $\Sigma_X$ is the covariance matrix of $X_{ij}$ and $\Sigma_{YX}$ is the covariance between $Y_{ij}$ and $X_{ij}$.
We decompose the observed data into independent within- and between components:

$$Z_{ij} = \mu_Z + Z^W_{ij} + Z^B_j\,, \tag{C.0.4}$$

$$\Sigma_Z = \Sigma^W_Z + \Sigma^B_Z\,, \tag{C.0.5}$$

where

$$\Sigma^W_Z = \begin{pmatrix} \Sigma^W_Y & \Sigma^W_{YX} \\ \Sigma^W_{XY} & \Sigma^W_X \end{pmatrix}\,, \tag{C.0.6}$$

$$\Sigma^B_Z = \begin{pmatrix} \Sigma^B_Y & \Sigma^B_{YX} \\ \Sigma^B_{XY} & \Sigma^B_X \end{pmatrix}\,, \tag{C.0.7}$$

and we have the following distribution for $Z^W_{ij}$ and $Z^B_j$:

$$Z^W_{ij} \sim \mathcal{N}(0, \Sigma^W)\,, \tag{C.0.8}$$

$$Z^B_j \sim \mathcal{N}(0, \Sigma^B)\,. \tag{C.0.9}$$

Recall the model equations from chapter MLSEM: **Level 1**

$$\begin{align} Y_{ij}^{W} &= \Lambda_Y^{W}\eta_{ij}^{W} + \varepsilon_{ij}^{W} , \tag{C.0.10} \\ X_{ij}^{W} &= \Lambda_X^{W}\xi_{ij}^{B} + \delta_{ij}^{W} , \tag{C.0.11} \\ \eta_{ij}^{W} &= B^{W}\eta_{ij}^{W} + \Gamma^{W}\xi_{ij}^{W} + \zeta_{ij}^{W} , \tag{C.0.12} \end{align}$$

**Level 2**

$$\begin{align} Y_j^{B} &= \Lambda_Y^{B}\eta_j^{B} + \varepsilon_j^{B} , \tag{C.0.13} \\ X_j^{B} &= \Lambda_X^{B}\xi_j^{B} + \delta_j^{B} , \tag{C.0.14} \\ \eta_j^{B} &= B^{B}\eta_j^{B} + \Gamma^{B}\xi_j^{B} + \zeta_j^{B} , \tag{C.0.15} \end{align}$$

What follows is a short derivation of $\Sigma_Y^{W}$ in terms of model parameters.

$$\begin{align} \Sigma_Y^{W} &= \mathrm{Var}(Y_{ij}^{W}) \tag{C.0.16} \\ &= \mathrm{Cov}(\Lambda_Y^{W}\eta_{ij}^{W} + \varepsilon_{ij}^{W} , \Lambda_Y^{W}\eta_{ij}^{W} + \varepsilon_{ij}^{W}) \tag{C.0.17} \\ &= \mathrm{Cov}(\Lambda_Y^{W}\eta_{ij}^{W} , \Lambda_Y^{W}\eta_{ij}^{W}) + \mathrm{Cov}(\varepsilon_{ij}^{W} , \varepsilon_{ij}^{W}) \tag{C.0.18} \\ &= \Lambda_Y^{W}\Sigma_\eta^{W}(\Lambda_Y^{W})^{T} + \Theta_\varepsilon^{W} . \tag{C.0.19} \end{align}$$

We substitute the left hand side of C.0.10 into C.0.16 to obtain C.0.17, then we use linearity of $\mathrm{Cov}(X , Y)$ and the fact that $\eta_{ij}^{W}$ and $\varepsilon_{ij}^{W}$ are independent to equate C.0.17 to C.0.18 and we use $\mathrm{Cov}(\varepsilon_{ij}^{W} , \varepsilon_{ij}^{W}) = \mathrm{Var}(\varepsilon_{ij}^{W}) = \Theta_\varepsilon^{W}$ and basic covariance algebra (specifically, to move $\Lambda_Y^{W}$ outside of the Cov function) to equate C.0.18 to C.0.19 (for information on the definition of $\Theta_\varepsilon^{W}$ or other model parameters, see table 2).

The derivations for $\Sigma_Y^{B}$, $\Sigma_X^{W}$ and $\Sigma_X^{B}$ are nearly identical and thus only the result will be stated:

$$\begin{align} \Sigma_Y^{W} &= \Lambda_Y^{W}\Sigma_\eta^{W}(\Lambda_Y^{W})^{T} + \Theta_\varepsilon^{W} , \tag{C.0.20} \\ \Sigma_Y^{B} &= \Lambda_Y^{B}\Sigma_\eta^{B}(\Lambda_Y^{B})^{T} + \Theta_\varepsilon^{B} , \tag{C.0.21} \\ \Sigma_X^{W} &= \Lambda_X^{W}\Phi^{W}(\Lambda_X^{W})^{T} + \Theta_\delta^{W} , \tag{C.0.22} \\ \Sigma_X^{B} &= \Lambda_X^{B}\Phi^{B}(\Lambda_X^{B})^{T} + \Theta_\delta^{B} . \tag{C.0.23} \end{align}$$

Note that the expressions for $\Sigma_Y^{W}$ and $\Sigma_Y^{B}$ contain the terms $\Sigma_\eta^{W}$ and $\Sigma_\eta^{W}$. These covariance matrices of $\eta_{ij}^{W}$ and $\eta_j^{B}$ are not parameters of the model, rather we will derive an expression for them using the within and between structural models. First, however, we will derive an expression for $\Sigma_{YX}^{W}$ and by similarity state the result for $\Sigma_{YX}^{B}$.

$$\begin{align} \Sigma_{YX}^{W} &= \mathrm{Cov}(Y_{ij}^{W} , X_{ij}^{W}) \tag{C.0.24} \\ &= \mathrm{Cov}(\Lambda_Y^{W}\eta_{ij}^{W} + \varepsilon_{ij}^{W} , \Lambda_X^{W}\xi_{ij}^{B} + \delta_{ij}^{W}) \tag{C.0.25} \\ &= \Lambda_Y^{W}\mathrm{Cov}(\eta_{ij}^{W} , \xi_{ij}^{W})(\Lambda_X^{W})^{T} \tag{C.0.26} \\ &= \Lambda_Y^{W}\Sigma_{\eta\,\xi}^{W}(\Lambda_X^{W})^{T} . \tag{C.0.27} \end{align}$$

We again substitute the left hand sides of C.0.10 and C.0.11 into the right hand side of C.0.24 to obtain C.0.25. Then by linearity of Cov, independence of $\varepsilon_{ij}^W$ and $\delta_{ij}^W$ and covariance algebra to obtain C.0.26. Again we encounter a term, labeled $\Sigma_{\eta\xi}^W$ in C.0.27, that is not a model parameter. We will derive an expression for this term in terms of model parameters.
Below, the results for $\Sigma_{YX}^W$ and $\Sigma_{XY}^B$ are stated:

$$\Sigma_{XY}^W = \Lambda_Y^W \, \Sigma_{\eta\,\xi}^W \, (\Lambda_X^W)^T \,, \tag{C.0.28}$$

$$\Sigma_{XY}^B = \Lambda_Y^B \, \Sigma_{\eta\,\xi}^B \, (\Lambda_X^B)^T \,. \tag{C.0.29}$$

It remains to find expressions for $\Sigma_\eta$ and $\Sigma_{\eta\,\xi}$ at both the within and between levels. We will show a derivation of $\Sigma_\eta^W$ and $\Sigma_{\eta\,\xi}^W$ and by similarly state the results for the between results.
We start with equationC.0.12 and derive the following using matrix algebra:

$$\begin{aligned} \eta_{ij}^W &= B^W \, \eta_{ij}^W + \Gamma^W \, \xi_{ij}^W + \zeta_{ij}^W \;\Rightarrow \\ (I - B^W) \, \eta_{ij}^W &= \Gamma^W \, \xi_{ij}^W + \zeta_{ij}^W \;\Rightarrow \\ \eta_{ij}^W &= (I - B^W)^{-1} \, (\Gamma^W \xi_{ij}^W + \zeta_{ij}^W) \end{aligned} \tag{C.0.30}$$

Here we assumed that $(I - B^W)$ is invertible. In appendix A, we show that this matrix is invertible if the directed graph consisting of only the structural relationships, i.e. we only keep the directed arrows between latent variables and disregard factor loadings and residual covariances, of the path diagram corresponding to the (within-level) SEM model is a Directed Acyclic Graph.
If we substitute the right hand side of the last line in C.0.30 into $\Sigma_{\eta\,\xi}^W = \text{Cov}(\eta_{ij}^W \,, \xi_{ij}^W \,)$, we get the following:

$$\Sigma_{\eta\,\xi}^W = \text{Cov}((I - B^W)^{-1} \, (\Gamma^W \xi_{ij}^W + \zeta_{ij}^W) \, \xi_{ij}^W \,) \tag{C.0.31}$$

$$= (I - B^W)^{-1} \, \Gamma^W \, \Phi^W \,. \tag{C.0.32}$$

We used linearity of Cov, independence of $\zeta_{ij}^W$ and $\xi_{ij}^W$ and covariance algebra to derive C.0.32 from C.0.31.

We now employ a similar strategy to derive $\Sigma_\eta^W$ by substituting the right hand side of the last line in C.0.30 into $\text{Cov}(\eta_{ij}^W \,, \eta_{ij}^W \,)$.

$$\Sigma_\eta^W = \text{Cov}((I - B^W)^{-1} \, (\Gamma^W \xi_{ij}^W + \zeta_{ij}^W) \,, (I - B^W)^{-1} \, (\Gamma^W \xi_{ij}^W + \zeta_{ij}^W) \,) \tag{C.0.33}$$

$$= (I - B^W)^{-1} \left[ \Gamma^W \, \Phi^W \, (\Gamma^W)^T + \Psi^W \right] ((I - B^W)^{-1})^T \tag{C.0.34}$$

We apply linearity of Cov, independence of $\xi_{ij}^W$ and $\zeta_{ij}^W$ to nullify the cross-terms and covariance algebra to move the matrices outside of the Cov function.

Finally, we replace the $\mathrm{Cov}(\xi_{ij}^{W}, \xi_{ij}^{W})$ and $\mathrm{Cov}(\zeta_{ij}^{W}, \zeta_{ij}^{W})$ with their respective model parameters, $\Phi^W$ and $\Psi^W$.

Now we state the results for both within and between components:

$$\Sigma_{\eta}^{W} = (I - B^{W})^{-1} \left[ \Gamma^{W} \, \Phi^{W} \, (\Gamma^{W})^{T} + \Psi^{W} \right] ((I - B^{W})^{-1})^{T} \,, \tag{C.0.35}$$

$$\Sigma_{\eta}^{B} = (I - B^{B})^{-1} \left[ \Gamma^{B} \, \Phi^{B} \, (\Gamma^{B})^{T} + \Psi^{B} \right] ((I - B^{B})^{-1})^{T} \,, \tag{C.0.36}$$

$$\Sigma_{\eta \, \xi}^{W} = (I - B^{W})^{-1} \, \Gamma^{W} \Phi^{W} \,, \tag{C.0.37}$$

$$\Sigma_{\eta \, \xi}^{B} = (I - B^{B})^{-1} \, \Gamma^{B} \Phi^{B} \,, \tag{C.0.38}$$

$$\tag{C.0.39}$$

If we now substitute equations C.0.35 and C.0.36 into equations C.0.20 and C.0.21, respectively, we get expressions for $\Sigma_Y^W$ and $\Sigma_Y^B$ in terms of model parameters. Similarly, if we substitute equations C.0.37 and C.0.38 into equations C.0.28 and C.0.29, respectively, we get expressions for $\Sigma_{YX}^W$ and $\Sigma_{YX}^B$ in terms of model parameters. We had already obtained expressions for $\Sigma_X^W$ and $\Sigma_X^B$ in terms of model parameters in equations C.0.22 and C.0.23.
Since these substitutions can get quite lengthy, we abbreviate it to:

$$\Sigma_Y^W = \Sigma_Y^W(\theta) \,, \tag{C.0.40}$$

$$\Sigma_Y^B = \Sigma_Y^B(\theta) \,, \tag{C.0.41}$$

$$\Sigma_X^W = \Sigma_X^W(\theta) \,, \tag{C.0.42}$$

$$\Sigma_X^B = \Sigma_X^B(\theta) \,, \tag{C.0.43}$$

$$\Sigma_{YX}^W = \Sigma_{YX}^W(\theta) \,, \tag{C.0.44}$$

$$\Sigma_{YX}^B = \Sigma_{YX}^B(\theta) \,, \tag{C.0.45}$$

$$\tag{C.0.46}$$

# D Code

## D.1 Preprocessing Data

```
import pandas as pd
import numpy as np
import scipy as sp
import scipy.stats as stats

import matplotlib.pyplot as plt
import matplotlib as mpl
import seaborn as sns


df = pd.read_excel(r"./Data/PRIME CL study Self-report 
    Final Dataset to Jesse_final.xlsx")

drop_cols =\
     ["Lecture", "BookHomework", "Collegerama", "Date_WMC1
         ", "Age", "Gender", "Gender_other", "Finished_S1",
           "Judgement_S1", "Judgement_S6"] +\
     df.columns[df.columns.str.contains("Duration")].
         to_list() +\
     df.columns[df.columns.str.contains("Recorded")].
         to_list()


df = df.drop(drop_cols, axis=1).set_index("AnalysisID")

survey_cols = df.columns[df.columns.str.contains(r"S[\d]"
    , regex=True)]
non_survey_cols = df.columns[~df.columns.str.contains(r"S
    [\d]", regex=True)]

col_df = survey_cols.str.extractall(r"(.*)_S(\d)|S(\d)_
    (.*)")
col_df.loc[col_df[0].isna(), [0, 1]] = col_df.loc[col_df
    [0].isna(), [3, 2]].to_numpy()

new_survey_cols = pd.MultiIndex.from_arrays(col_df[[0,
    1]].to_numpy().transpose(), names=['var', 'survey'])

survey_data = df[survey_cols].copy()
survey_data.columns = new_survey_cols
```

```
survey_data_stacked = survey_data.stack(level='survey',
    future_stack=True).reset_index().set_index("AnalysisID
    ")

non_survey_data = df[non_survey_cols].copy()

joined_df = pd.merge(
     survey_data_stacked,
     non_survey_data,
     how='left',
     left_index=True,
     right_index=True
     )\
     .reset_index()

joined_df['Efficacy'] = joined_df['Efficacy'] / 10.0

def log_transform(x, N=10):
     p = (x + 0.5) / (N + 1)

     return np.log(p / (1-p))



tCols = ["Efficacy", "Difficulty", "MentalEffort", "ICL",
     "GCL", "ECL"]

joined_df[tCols] = log_transform(joined_df[tCols])


def center_group_means(df: pd.DataFrame, group_col: str,
    cols: list):
     df = df[[group_col] + cols].copy()
     group_means = df.groupby(group_col).mean()
     left, right = df.set_index(group_col).align(
         group_means)

     return (left - right).reset_index(), right.
         reset_index()


group_mean_cols = ["Efficacy", "MentalEffort", "
    Difficulty", "ECL", "GCL", "ICL"]

group_centered, group_means = center_group_means(
    joined_df, "AnalysisID", group_mean_cols)
```

```
new_cols = {
    f"{col}_c": group_centered[col]
    for col in group_mean_cols
}

final_df = joined_df.assign(**new_cols)
final_df = pd.merge(final_df, group_means.iloc[:, 1:],
   how='left', left_index=True, right_index=True,
   suffixes=['', '_mean'])

filter_list_ids = final_df\
    .dropna(axis=0, how='any')\
    .groupby("AnalysisID", as_index=False)\
    .count()\
    .where(lambda x: x.survey >= 3)\
    .dropna(axis=0, how='any')\
    ["AnalysisID"]


final_df\
    .loc[final_df["AnalysisID"].isin(filter_list_ids),
       :]\
    .dropna(axis=0, how='any')\
    .to_csv("./Data/transformed_final_dataset.csv", index
       =False)
```

## D.2 Simulate Data

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

import inspect
import os


# Sizes
n_clus = 40 # number of clusters
clus_size = 5 # Cluster size

default_params = {
    # Error / Random component variances
    'sigma_eps': [1.5, 1.5],
```

```python
    'sigma_u': [0.7, 1.2],

    # Covariate variances
    'sigma_x': [1.5, 1.0],

    # Regression params
    # Within
    'w': [1.5, -0.2],

    # Between
    'b': [0.4, 0.0],

    # endogenous within
    'lam': [[1.0, 0.0],
            [0.3, 1.0]],

    # Fixed effects
    'eta0': [0.4, 1.0],

    # Cluster size params
    'n_clus': n_clus,
    'clus_size': clus_size
}


def parse_base_case(label, item) -> dict:
    return {"label": label, "value": item}

def parse_array(lst: list, prefix="array_") -> list:
    res = []
    i = 0
    for item in lst:
        label = f"{prefix}{i}"
        if isinstance(item, list):
            res += parse_array(item, label)
        else:
            res.append(parse_base_case(label, item))

        i += 1

    return res

def parse_params(params:dict) -> list:
    res = []
    for label, item in params.items():
        if isinstance(item, list):
            res += parse_array(item, label)
        else:
            res.append(parse_base_case(label, item))

    return res

def params_to_df(params: dict) -> pd.DataFrame:
    parsed_params = parse_params(params)

    df = pd.DataFrame.from_records(parsed_params)

    return df


class SimulationModel(object):

    def __init__(self, default_params: dict=dict()):
        self.default_params = default_params

    def transform_params(self, params: dict) -> dict:
        return params

    def model(self, *args, **kwargs) -> pd.DataFrame:
        ...

    def get_params(self, **kwargs):
        p = dict(**self.default_params)
        p.update(kwargs)

        return p

    def simulate_data(self, **kwargs) -> pd.DataFrame:
        params = self.get_params(**kwargs)
        working_params = self.transform_params(params)

        df = self.model(**working_params)

        return df


class SimulationExec(object):

    def __init__(self, folder_path=r”./simulation”):
        self.folder_path = folder_path

        # Checks if root folder exists, otherwise raises
```

```
            error
        if not os.path.exists(self.folder_path):
            raise OSError(f"Root␣folder:␣'{self.
                folder_path}'␣does␣not␣exist.␣Make␣sure␣
                the␣provided␣root␣folder␣exists.")

    def init_dirs(self, sim_id):
        if os.path.exists(f"{self.folder_path}/{sim_id}")
            :
            raise OSError(f"Provided␣sim_id:␣'{sim_id}'␣
                is␣already␣used.␣Provide␣an␣unused␣value."
                )
        else:
            # Makes all required dirs
            os.makedirs(f"{self.folder_path}/{sim_id}/
                data")

    def run_simulation(self, model:SimulationModel,
       sim_id="mysim", n_sims=10):
        # Initialize required folders
        self.init_dirs(sim_id)

        folder_path = f"{self.folder_path}/{sim_id}"

        run_params = model.get_params()
        run_param_df = params_to_df(run_params)
        run_param_df.to_csv(f"{folder_path}/params.csv",
            header=True)

        for i in range(n_sims):
            df = model.simulate_data()
            df.to_csv(f"{folder_path}/data/sim-{i}.csv",
                header=True)



class MyModel(SimulationModel):

    def __init__(self, params: dict=dict()):
        super().__init__(params)

    def transform_params(self, params: dict) -> dict:
        working_params = dict(**params)

        working_params['sigma_u'] = np.diag(params['
            sigma_u'])
        working_params['sigma_eps'] = np.diag(params['
            sigma_eps'])

        working_params['sigma_x'] = np.array(params['
            sigma_x'])
        working_params['lam'] = np.array(params['lam'])
        working_params['w'] = np.array(params['w'])
        working_params['b'] = np.array(params['b'])
        working_params['eta0'] = np.array(params['eta0'])

        return working_params

    def model(self, eta0, lam, w, b, sigma_x, sigma_u,
        sigma_eps, n_clus, clus_size) -> pd.DataFrame:

        # Cluster indices
        clus_id = np.arange(0, n_clus).repeat(clus_size)

        # Sample covariates / predictors
        x1 = sigma_x[0] * np.random.randn(n_clus *
            clus_size)
        x2 = sigma_x[1] * np.random.randn(n_clus).repeat(
            clus_size)

        # Sample random comps / residuals
        eps_ij = np.random.randn(n_clus * clus_size, 2) @
             sigma_eps

        u_j = np.random.randn(n_clus, 2).repeat(clus_size
            , axis=0) @ sigma_u

        # # Random intercept
        eta_j = eta0 + x2.reshape(-1, 1) @ b.reshape(1,
            -1) + u_j

        # Response variable
        y_ij = eta_j + x1.reshape(-1, 1) @ w.reshape(1,
            -1) + eps_ij @ lam

        # Create dataframe holding simulation data
        df = pd.DataFrame({
             'cluster_id': clus_id,
             'y1': y_ij[:, 0],
             'y2': y_ij[:, 1],
             'x1': x1,
             'x2': x2
        })

        return df



param_variations = [
    {
        'sigma_u': [1.0, 1.0],
        'name': 'param-1'
    },
    {
        'sigma_u': [0.2, 1.0],
        'name': 'param-2'
    },
    {
        'sigma_u': [1.0, 0.2],
        'name': 'param-3'
    },
    {
        'sigma_u': [0.2, 0.2],
        'name': 'param-4'
    }
]


executor = SimulationExec(r"./simulation/run-5")

n_sims = 50
for p_var in param_variations:
    p = dict(**default_params)
    p['sigma_u'] = p_var['sigma_u']

    model = MyModel(p)
    executor.run_simulation(model, sim_id=p_var['name'],
        n_sims=n_sims)
```

## D.3 Simulation Study

```
library(tidyverse)
library(lavaan)
library(blavaan)
library(readr)
library(ggplot2)
```

```
model <- '
  level: within
  # Regression
    y1 ~ w1*x1 + lam1*y2
    y2 ~ w2*x1

  # Variances
    y1 ~~ e1*y1
    y2 ~~ e2*y2

  level: between
  # Regressions
    y1 ~ b1*x2
    y2 ~ b2*x2

   # Intercepts
    y1 ~ eta1*1
    y2 ~ eta2*1

  # Variances
    y1 ~~ u1*y1
    y2 ~~ u2*y2
'

fitModel <- function(file_path, prior) {
  # Read in data frame
  data <- read_csv(
    file_path,
    col_types = cols(cluster_id = col_integer())
  )

  fit <- bsem(
    model = model,
    data = data,
    cluster=”cluster_id”,
    dp=dpriors(psi=prior)
  )

  return(fit)
}


extractResults <- function(fit) {
  median <- blavInspect(fit, ”postmedian”)
  mode <- blavInspect(fit, ”postmode”)
  mean <- blavInspect(fit, ”postmean”)
  rhat <- blavInspect(fit, "rhat")
  neff <- blavInspect(fit, "neff")

  median$var <- "median"
  mode$var <- "mode"
  mean$var <- "mean"
  rhat$var <- "rhat"
  neff$var <- "neff"

  res <- list(
    median,
    mode,
    mean,
    neff,
    rhat
  )

  #res <- do.call(rbind, res)

  return(res)
}

processFile <- function(file_path, prior) {
  fit <- fitModel(file_path, prior)
  res <- extractResults(fit)

  return(res)
}


processFiles <- function(files, prior) {
  lst <- list()
  for (k in 1:length(files)) {
    res_k <- processFile(files[[k]], prior)
    lst <- append(lst, res_k)
  }

  lst <- do.call(rbind, lst)
  df_res <- data.frame(lst)
  for (c_name in colnames(df_res)) {
    df_res[c_name] <- unlist(df_res[c_name])
  }

  return(df_res)
}
```

```
getFiles <- function(wildcard_path) {
  files <- Sys.glob(wildcard_path)
  return(files)
}


processFolder <- function(folder_path, prior) {
  wildcard_path <- paste(folder_path, "/data/*", sep="")
  files <- getFiles(wildcard_path)

  df_res <- processFiles(files, prior)

  return(df_res)
}


paths = list(
  "param-1",
  "param-2",
  "param-3",
  "param-4"
)

priors = list(
  "[sd]",
  "[var]",
  "[prec]"
)

prior_form <- "gamma(1,.5)"

out_folder <- "../notebooks/simulation-out/run-5/"
in_folder <- "../notebooks/simulation/run-5/"

for (path in paths) {
  fp <- paste(in_folder, path, sep="")

  for (k in 1:length(priors)) {
    prior <- paste(prior_form, priors[[k]], sep="")

    df_res <- processFolder(fp, prior)
    df_res$simId <- path
    df_res$prior <- priors[[k]]

    out_file <- paste(out_folder, path,"-prior-", k, ".
        csv", sep="")

    write.csv(df_res, out_file)
  }
}

#fit <- fitModel("../notebooks/simulation/run-2/param-4/
    data/sim-0.csv", prior="gamma(1,.5)[prec]")
#summary(fit)
```

## D.4 Simulation Result Plots

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

import os
from typing import Dict, Callable


plt.style.use("ggplot")

RUN_ID = 'run-5'

ALIASES = {
    'w1': 'w0',
    'w2': 'w1',
    'lam1': 'lam10',
    'e1': 'sigma_eps0',
    'e2': 'sigma_eps1',
    'b1': 'b0',
    'b2': 'b1',
    'eta1': 'eta00',
    'eta2': 'eta01',
    'u1': 'sigma_u0',
    'u2': 'sigma_u1'
}

REVERSE_ALIASES = dict()
for key, val in ALIASES.items():
    REVERSE_ALIASES[val] = key


def load_simulation_ouput(run_id: str):
    folder_path = fr"./simulation-out/{run_id}"

    dfs = []
    for file_name in os.listdir(folder_path):
        df_k = pd.read_csv(f"{folder_path}/{file_name}")
        dfs.append(df_k)

    df_ = pd.concat(dfs, axis=0)\
        .drop("Unnamed:␣0", axis=1)\
        .reset_index(drop=True)

    df_ = df_\
        .assign(new_idx=df_.index.to_numpy() // df_["var"
            ].nunique())\
        .pivot(index="new_idx", columns="var")\
        .swaplevel(0, 1, axis=1)

    return df_


def load_params(run_id: str):
    folder_path = fr"./simulation/{run_id}"

    dfs_ = []
    for path in os.listdir(folder_path):
        df_ = pd.read_csv(f"{folder_path}/{path}/params.
            csv", usecols=[1, 2])
        df_["simId"] = path
        dfs_.append(df_)

    df = pd.concat(dfs_, axis=0)
    piv_df = df.pivot(columns='label', index='simId',
       values='value').rename(REVERSE_ALIASES, axis=1)

    var_cols = ["e1", "e2", "u1", "u2"]
    piv_df[var_cols] = piv_df[var_cols]**2

    return piv_df



def is_outlier(arr, scale=2.5):
    arr = np.array(arr)

    lhinge = np.quantile(arr, 0.25)
    median = np.median(arr)
    uhinge = np.quantile(arr, 0.75)

    iqr = uhinge - lhinge

    dev = np.abs(arr - median) / iqr

    return (dev >= scale)


def check_rhat(df: pd.DataFrame, lb=0.995, ub=1.005):
    '''
    This function expects a DataFrame containing the rhat
         statistics for all the variables.
    It checks if the rhat lies in an acceptable region,
        so the estimates can be seen as reliable.
    '''
    bool_df = (lb < df) & (df < ub)
    bool_idx = bool_df.all(1)

    return bool_idx


def check_ess(df: pd.DataFrame, min_val=100):
    '''
    This function expects a DataFrame containing the ess
        for all the variables.
    It checks if the ess is high enough to interpret the
        posterior estimates as representative.
    '''
    bool_df = df > min_val
    bool_idx = bool_df.all(1)

    return bool_idx


def check_reliable(df: pd.DataFrame):
    '''
    This function expects a DataFrame containing all
        simulation data. That is, all the measures
    gathered from the fit objects in R.
    It checks if the rhat of the variables are acceptable
         and the effective sample size is large enough.
    Each simulation corresponds to a single row, if one
        variable does not satisfy the rhat and ess
        conditions, the entire
    row is tossed.
    It returns a copy of the filtered DataFrame.
    '''
    ess_min = 100
    rhat_min = 0.995
    rhat_max = 1.005

    df_rhat = df["rhat"].iloc[:, :-2]
    df_ess = df["neff"].iloc[:, :-2]

    rhat_bool = check_rhat(df_rhat, rhat_min, rhat_max)
    ess_bool = check_ess(df_ess, ess_min)

    reliable_bool = rhat_bool & ess_bool

    return df.loc[reliable_bool, :].copy()


def compute_diff(estimates_df: pd.DataFrame, params_df:
   pd.DataFrame, col: str) -> np.ndarray:
    estimates = estimates_df[col].to_numpy()
    true_vals = params_df.loc[estimates_df["simId"], col
        ].to_numpy()

    bias = estimates - true_vals

    return bias

def compute_diffs(estimates_df: pd.DataFrame, params_df:
   pd.DataFrame, cols: list[str]) -> pd.DataFrame:
    data = {
        col: compute_diff(estimates_df, params_df, col)
        for col in cols
    }
    df = pd.DataFrame(data)


    df["simId"] = estimates_df["simId"]
    df["prior"] = estimates_df["prior"]
    df.index = estimates_df.index

    return df


def compute_biases(estimates_df: pd.DataFrame, params_df:
    pd.DataFrame, cols: list[str]) -> pd.DataFrame:
    diffs_df = compute_diffs(estimates_df, params_df,
        cols)

    biases_df = diffs_df.groupby(["prior", "simId"],
        as_index=False).mean()

    return biases_df


def compute_rmse(estimates_df: pd.DataFrame, params_df:
   pd.DataFrame, cols: list[str]) -> pd.DataFrame:
    df = compute_diffs(estimates_df, params_df, cols)

    df[cols] = df[cols]**2

    rmse_df = df.groupby(["prior", "simId"], as_index=
        False).mean()
    rmse_df[cols] = np.sqrt(rmse_df[cols])

    return rmse_df


def facet_boxplot(df: pd.DataFrame, var_name: str, col:
   str, hue: str):
    _df = df.loc[~is_outlier(df[var_name]), :]

    grid = sns.FacetGrid(data=_df, col=col, col_wrap=2,
        legend_out=True)
    grid.map_dataframe(sns.boxplot, x=var_name, hue=hue)
    grid.add_legend()


latex_var = {
    "u1": r"\zeta^{1,-B}",
    "u2": r"\zeta^{2,-B}"
}



def plot_biases(biases_df: pd.DataFrame, params_df: pd.
   DataFrame, var:str):
    temp_ = pd.merge(
         biases_df,
         params_df[["u1", "u2"]].map(lambda x: f"{x:.2f}")
             ,
         left_on="simId",
         right_index=True,
        suffixes=["", "_param"]
    )

    g = sns.FacetGrid(
        data=temp_,
        row="u1_param", col="u2_param", hue="prior",
        height=4, aspect=1
    )
    g.map_dataframe(sns.barplot, y=var, x="prior")
    g.set_ylabels(r"$\text{Bias:}~" + latex_var[var] + r"
      $")
    g.add_legend()

    return g


def plot_rmse(rmse_df: pd.DataFrame, params_df: pd.
   DataFrame, var:str):
    temp_ = pd.merge(
        rmse_df,
        params_df[["u1", "u2"]].map(lambda x: f"{x:.2f}")
            ,
        left_on="simId",
        right_index=True,
        suffixes=["", "_param"]
    )

    g = sns.FacetGrid(
        data=temp_,
        row="u1_param", col="u2_param", hue="prior",
        height=4, aspect=1
    )
    g.map_dataframe(sns.barplot, y=var, x="prior")
    g.set_ylabels(r"$\text{RMSE:}~" + latex_var[var] + r"
      $")
    g.add_legend()

    return g



df_params = load_params(RUN_ID)

df_ = load_simulation_ouput(RUN_ID)
df = check_reliable(df_)
```

```
df_mean = df["mean"].copy()


cols = list(ALIASES.keys())
biases_df = compute_biases(df_mean, df_params, cols)
rmse_df = compute_rmse(df_mean, df_params, cols)

df_mean.loc[is_outlier(df_mean["u2"]), "u2"]

g1 = plot_biases(biases_df, df_params, "u1")
g2 = plot_biases(biases_df, df_params, "u2")

g3 = plot_rmse(rmse_df, df_params, "u1")
g4 = plot_rmse(rmse_df, df_params, "u2")


latex_labels = {
    'u1_param␣=␣1.00␣|␣u2_param␣=␣1.00': r"$\zeta^{1,␣B}␣
       =␣1.00␣|␣\zeta^{2,␣B}␣=␣1.00$",
    'u1_param␣=␣1.00␣|␣u2_param␣=␣0.04': r"$\zeta^{1,␣B}␣
       =␣1.00␣|␣\zeta^{2,␣B}␣=␣0.04$",
    'u1_param␣=␣0.04␣|␣u2_param␣=␣1.00': r"$\zeta^{1,␣B}␣
       =␣0.04␣|␣\zeta^{2,␣B}␣=␣1.00$",
    'u1_param␣=␣0.04␣|␣u2_param␣=␣0.04': r"$\zeta^{1,␣B}␣
       =␣0.04␣|␣\zeta^{2,␣B}␣=␣0.04$"
}


for g in [g1, g2, g3, g4]:
    for ax in g.axes.flatten():
        current_title = ax.title.get_text()
        ax.set_title(latex_labels[current_title])

g1.savefig("./figures/bias_u1.png")
g2.savefig("./figures/bias_u2.png")

g3.savefig("./figures/rmse_u1.png")
g4.savefig("./figures/rmse_u2.png")
```

## D.5 PRIME research Model Fitting

```
library(tidyverse)
library(lavaan)
library(blavaan)
library(readr)
library(xtable)
library(dplyr)


# Read in data frame from CSV file containing
    preprocessed data
data <- read_csv(
  "C:/Users/31620/OneDrive␣−␣GE/BEP/notebooks/Data/
      transformed_final_dataset.csv",
  col_types = cols(
    survey = col_integer()
  )
)

# Apply filter to data to filter out students with no
    within level variance (same score across surveys)
data <- filter(data, !(AnalysisID %in% c("C39", "A44", "
    A38", "C58"))) # New problem id: no within variance


# Model definition
model <- '
␣␣level:␣within
␣␣␣␣MentalEffort␣~␣GCL␣+␣ECL␣+␣Efficacy␣+␣ICL
␣␣␣␣GCL␣~␣Efficacy


␣␣level:␣between
␣␣␣␣MentalEffort␣~␣PriorKnowledge␣+␣Efficacy␣+␣
    Globalscore_WMC1␣+␣GCL␣+␣ECL␣+␣ICL
␣␣␣␣ICL␣~␣PriorKnowledge
␣␣␣␣ECL␣~␣Globalscore_WMC1

'



# Fit model using bsem blavaan function. Default prior
    for variances is based on simulation results
bfit <- bsem(model=model, data=data, dp=dpriors(psi="
    gamma(1,.5)[sd]"), cluster="AnalysisID", mcmcfile =
    TRUE)


## Summary extraction
# Extract posterior samples
samples_list <- blavInspect(bfit, "mcmc")
samples_mat <- do.call(rbind, samples_list)
```

```
# Get column names
posterior_names <- colnames(samples_mat)

# Classify type and level
classify_param <- function(name) {
  level <- if (grepl("\\.lbetween$", name)) "between"
      else "within"
  if (grepl("~1", name)) {
    type <- "intercept"
    clean <- gsub("~1(\\.lbetween)?", "", name)
    label <- paste0(clean, "␣intercept␣(", level, ")")
  } else if (grepl("~~", name)) {
    vars <- gsub("\\.lbetween", "", name)
    label <- paste0(vars, "␣residual␣variance␣(", level,
       ")")
    type <- "residual␣variance"
  } else if (grepl("~", name)) {
    parts <- strsplit(gsub("\\.lbetween", "", name), "~")
       [[1]]
    label <- paste0(parts[1], "␣~␣", parts[2], "␣(",
       level, ")")
    type <- "regression"
  } else {
    type <- "unknown"
    label <- name
  }
  return(c(type = type, level = level, label = label))
}

# Apply classification
classified <- t(sapply(posterior_names, classify_param))
mapping_df <- data.frame(
  Parameter = posterior_names,
  Type = classified[, "type"],
  Level = classified[, "level"],
  Readable = classified[, "label"],
  stringsAsFactors = FALSE
)

# Compute posterior summaries
summaries <- lapply(mapping_df$Parameter, function(param)
    {
  vec <- samples_mat[, param]
  data.frame(
    Parameter = param,
    Mean = mean(vec),
    SD = sd(vec),
    CI_lower = quantile(vec, 0.025),
    CI_upper = quantile(vec, 0.975),
    stringsAsFactors = FALSE
  )
})

results_df <- do.call(rbind, summaries)

# Merge with readable labels
results_df <- left_join(results_df, mapping_df, by = "
   Parameter")

#xt <- xtable(
#  results_df[, c("Readable", "Mean", "SD", "CI_lower", "
   CI_upper", "Type", "Level")],
#  digits = 3,
#  caption = "Unstandardized Posterior Estimates with
   95\\% Credible Intervals"
#)
```

### D.6 Prior Plots

```
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt
import plotly.graph_objects as go


def gamma(x, alpha, beta):
    return stats.gamma.pdf(x, alpha, scale=1/beta)


def gamma_sd(x, alpha, beta):
    return gamma(np.sqrt(x), alpha, beta) / (2 * np.sqrt(
        x))


def gamma_prec(x, alpha, beta):
    return (1 / (x**2)) * gamma(1/x, alpha, beta)


def generate_plot(x, alpha, beta, ax: plt.Axes):
    ax.plot(x, gamma(x, alpha, beta), label='var')
    ax.plot(x, gamma_sd(x, alpha, beta), label='sd')
    ax.plot(x, gamma_prec(x, alpha, beta), label='
        precision')
    ax.grid(True)
    ax.legend()
    ax.set_title(f"alpha = {alpha:.2f} beta = {beta:.2f}"
        )


alphas = [0.5, 1, 3]
betas = [0.5, 1, 3]
x = np.linspace(0.01, 10, 200)

fig = plt.figure(figsize=(12, 12))
axes = fig.subplots(3, 3)


for i, alpha in enumerate(alphas):
    for j, beta in enumerate(betas):
        ax = axes[i][j]

        generate_plot(x, alpha, beta, ax)


plt.savefig('./prior_plots')
```

# References

[1] Zitzmann et al. "Prior Specification for More Stable Bayesian Estimation of Multilevel Latent Variable Models in Small Samples: A Comparative Investigation of Two Different Approaches". In: *Frontiers in Psychology* (2020).

[2] K. A. Bollen. *Structural Equations with Latent Variables.* Wiley, 1989.

[3] A. C. Brouwer. "Structural Equation Modeling: Explained and applied to educational science". Available at the TU Delft Library. 2021.

[4] H. de Bruin Anique B. et al. "Synthesizing Cognitive Load and Self-regulation Theory: a Theoretical Framework and Research Agenda". In: *Educational Psychology Review* 32.4 (2020), pp. 903–915. DOI: 10.1007/s10648-020-09576-4.

[5] Juan Cristóbal Castro-Alonso, Paul Ayres, and John Sweller. "Instructional visualizations, cognitive load theory, and visuospatial processing". In: *Visuospatial Processing for Education in Health and Natural Sciences.* Ed. by Crist'obal Castro-Alonso Juan. Advances in Experimental Medicine and Biology. Springer, 2019, pp. 111–143. DOI: 10.1007/978-3-030-20969-8_5.

[6] Sarah Depaoli. "A Bayesian Approach to Multilevel Structural Equation Modeling With Continious and Dichotomous Outcomes". In: *Structural Equation Modeling: A Multidisciplinary Journal, 22, 327-351* (2015).

[7] Sarah Depaoli. *Bayesian structural equation modeling.* The Guilford Press, 2021.

[8] S. van Erp and W. J. Browne. "Bayesian multilevel structural equation modeling: An investigation into robust prior distributions for the doubly latent categorical model". In: *Structural Equation Modelling: A Multidisciplinary Journal* (2021).

[9] J. Fox. *Applied Regression Analysis & Generalized Linear Models.* Sage, 2016.

[10] G. Grimmet and D. Welsh. *Probability: An Introduction.* Oxford University Press, 2014.

[11] J. J. Hox. *Mulitlevel Analysis Techniques and Applications.* Routledge, 2010.

[12] J. Leppink et al. "Development of an instrument for measuring different types of cognitive load". In: *Behavior Research Methods* 45.4 (2013), pp. 1058–1072. DOI: 10.3758/s13428-013-0334-1.

[13] J. Leppink et al. "Effects of pairs of problems and examples on task performance and different types of cognitive load". In: *Learning and Instruction* 30 (2014), pp. 32–42. DOI: 10.1016/j.learninstruc.2013.11.001.

[14] R. Levy and R. J. Mislevy. *Bayesian Psychometric Modeling.* Chapman & Hall, 2016.

[15] B. Muthen and T. Asparouhov. “Bayesian estimation of single and multilevel models with latent variable interactions”. In: *Structural Equation Modeling: A Multidisciplinary Journal* (2020).

[16] B. O. Muthen. “Multilevel Covariance Structure Analysis”. In: *Sociological Methods and Research, 22, 376-398* (1994).

[17] Y. Rosseel. “Evaluating the Observed Log-Likelihood Function in Two-Level Structural Equation Modeling with Missing Data: From Formulas to R Code”. In: *Psych* (2021).

[18] Shayle R Searle, George Casella, and Charles E McCulloch. *Variance Components*. New York: Wiley, 1992.