Bayesian psychometric modeling and blavaan

Ed Merkle

IMPS 2025 Short Course

Acknowledgments

- *blavaan* and related research have been funded by Institute of Education Sciences Grant R305D210044. The software has seen many contributors and collaborators over time! It would not be possible without Yves Rosseel.

Slides & More

https://ecmerkle.github.io/blavaan/articles/resources.html

Introduction

Introduction

- Goals of the course
  - Part 1: Overview and major ideas underlying Bayesian latent variable models
  - Part 2: See how these ideas work in practice
  - Part 3: Consider Gibbs samplers
  - Part 4: Consider Hamiltonian samplers
  - Conclude with recommendations, workflows, Q&A

Introduction

- Some themes
  - Emphasize the "raw data" part of the models
  - Model some real data
  - Provide intuition underlying popular MCMC methods
  - Include code, so that you can work through concepts in the future

Introduction

- What is not covered: R background, exotic psychometrics models.
- And much more can be done with model checking and criticism. We are time limited.
- I apologize now for discussing what you already know, and for assuming you know what you don't know!

Part 1: Overview

Model Overview

Model Overview

- In psychometrics and education, the latent variable in "latent variable models" is typically a person's unobserved trait.
- Let's call the latent variable η.
- While we cannot observe η directly, we can observe some other variable that serves as a proxy for η. Let's call that other variable y.

Linear

- Linear equations are everywhere in statistics. So too in psychometrics.
- For traditional models, we assume that our observed variable is a noisy, linear function of η.
- For now, let's write it as

$$y = \beta_0 + \beta_1 \eta + e$$

Linear

$$y = \beta_0 + \beta_1 \eta + e$$

- We left out subscripts. We could say that this is the model of a single person on a single observed variable.
- If η were observed, this would be a regression model.
- This is a model of one observed variable, but we typically have multiple observed variables per person.

Subscripts

Let's add subscripts for person i and observed variable j

$$y_{ij} = \beta_{0j} + \beta_{1j}\eta_i + e_{ij}$$

- The subscripts on β_0 and β_1 show that the intercept and slope are unique to an observed variable j.

Matrix

- It is customary to collect all the observed variables for person i in one vector. And to collect the intercepts for observed variables in one vector, and similarly for the slopes. Then the j subscripts disappear:

$$\boldsymbol{y}_i = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 \eta_i + \boldsymbol{e}_i$$

  where bold represents a vector (or a matrix, though no matrices are on this slide)

Matrix

- We may also be interested in measuring multiple latent traits per person. In this case, we also have a vector for η:

$$\boldsymbol{y}_i = \boldsymbol{\beta}_0 + \boldsymbol{\beta}_1 \boldsymbol{\eta}_i + \boldsymbol{e}_i$$

- Now $\boldsymbol{\beta}_1$ becomes a matrix.

Models

$$\beta_0 + \beta_1 \boldsymbol{\eta}_i + \boldsymbol{e}_i$$

- From this simple linear equation, we can obtain many traditional psychometric models:
  - Observed variables are continuous: Factor analysis, SEM.
  - Observed variables are binary or ordinal: 2-parameter logistic model, graded response model, generalized partial credit model.
  - In the latter case, the linear equation does not directly predict the observed variables. The linear equation predicts (a function of) the probability that a particular person assumes a particular category of a particular variable. The $\boldsymbol{e}_i$ term disappears or has fixed variance, depending on how the model is written.

Models

- The previous slide implies that β_0 and β_1 have different names in different situations. (and also different Greek letters!)
  - β_0: Intercept, mean, difficulty, easiness
  - β_1: Loading, slope, discrimination
- We have also ignored the fact that one latent variable can be predictive of a second latent variable. Structural Equation Models help us here, and they require a second linear equation.

SEM

- The *structural* equation of SEM:

$$\boldsymbol{\eta}_i = \boldsymbol{\alpha}_0 + \boldsymbol{\alpha}_1 \boldsymbol{\eta}_i + \boldsymbol{\zeta}_i$$

- This looks confusing because $\boldsymbol{\eta}_i$ appears on both sides of the equation.
- The key is to realize that no single element of $\boldsymbol{\eta}_i$ can predict itself: the diagonal of $\boldsymbol{\alpha}_1$ must equal $\mathbf{0}$. Each element of $\boldsymbol{\eta}_i$ can predict other elements, making this just another linear equation.

SEM

- Now let's change to the "LISREL" notation that is often used in SEM.

$$\boldsymbol{y}_i = \boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\eta}_i + \boldsymbol{\epsilon}_i \tag{1}$$
$$\boldsymbol{\eta}_i = \boldsymbol{\alpha} + \boldsymbol{B}\boldsymbol{\eta}_i + \boldsymbol{\zeta}_i \tag{2}$$

- Residual distributions:

$$\boldsymbol{\epsilon}_i \sim \mathrm{N}_p(\mathbf{0}, \boldsymbol{\Theta}) \tag{3}$$
$$\boldsymbol{\zeta}_i \sim \mathrm{N}_m(\mathbf{0}, \boldsymbol{\Psi}) \tag{4}$$

- Typically, the arrows in path diagrams are nonzero entries of $\boldsymbol{\Lambda}$ and of $\boldsymbol{B}$.

SEM

- Our equations so far all involve the latent variables $\boldsymbol{\eta}_i$. Those have an i subscript, and i goes from 1 to N (lots of people in the dataset). So it amounts to hundreds or thousands of additional parameters.
- Traditionally, we marginalize out the $\boldsymbol{\eta}_i$ to make model estimation easier. This is not absolutely necessary in Bayesian modeling, and it can lead to some differences in model estimation speed and efficiency.
- We will return to these ideas in the afternoon, but for now we will just look at the different forms of the model.

Conditional

- Conditional on the $\boldsymbol{\eta}_i$, we have

$$\boldsymbol{y}_i \mid \boldsymbol{\eta}_i \sim \mathsf{N}(\boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\eta}_i, \boldsymbol{\Theta}) \tag{5}$$

$$\boldsymbol{\eta}_i \sim \mathsf{N}((\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\alpha}, (\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Psi}(\boldsymbol{I} - \boldsymbol{B}')^{-1}) \tag{6}$$

- Marginalizing out the $\boldsymbol{\eta}_i$, we have

$$\boldsymbol{y}_i \sim \mathsf{N}(\boldsymbol{\nu} + \boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\alpha}, \boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Psi}(\boldsymbol{I} - \boldsymbol{B}')^{-1}\boldsymbol{\Lambda}' + \boldsymbol{\Theta}), \tag{7}$$

Two-Level

- We can also consider extensions to two-level SEM. This covers situations where, e.g., the people are nested in schools.
- It is sometimes helpful to think of this as a three-level model: observed variables nested in people nested in schools.
- The models become more difficult to estimate because we account for correlations between people who attend the same school.

Two-Level

- SEM with random intercepts, for person i in school (cluster) k:

$$\boldsymbol{y}_{ik} = \boldsymbol{\nu}_k + \boldsymbol{\Lambda}\boldsymbol{\eta}_{ik} + \boldsymbol{\epsilon}_{ik} \tag{8}$$

$$\boldsymbol{\eta}_{ik} = \boldsymbol{\alpha} + \boldsymbol{B}\boldsymbol{\eta}_{ik} + \boldsymbol{\zeta}_{ik} \tag{9}$$

- Notice the k subscript on $\boldsymbol{\nu}$. Each school has its own intercept. And the intercept is a vector (one intercept per observed variable).

Two-Level

- From a traditional multilevel modeling point of view, we could assign

$$\boldsymbol{\nu}_k \sim \mathrm{N}(\boldsymbol{\nu}^c, \boldsymbol{\Sigma}^c)$$

  where the c superscript signifies a school-level parameter.

- But this can lead to many extra parameters. For example, if there are 9 observed variables per person, then $\boldsymbol{\Sigma}^c$ contains 45 free parameters.

- To reduce the number of free parameters, we can specify a separate SEM for the $\boldsymbol{\nu}_k$. That is, school latent variables predict the $\boldsymbol{\nu}_k$.

Two-Level

- Altogether, the notation becomes a burden, but it is like we have the same model twice. First, the usual model, then the model of school intercepts (c superscripts):

$$\boldsymbol{y}_{ik} = \boldsymbol{\nu}_k + \boldsymbol{\Lambda}\boldsymbol{\eta}_{ik} + \boldsymbol{\epsilon}_{ik} \tag{10}$$

$$\boldsymbol{\eta}_{ik} = \boldsymbol{\alpha} + \boldsymbol{B}\boldsymbol{\eta}_{ik} + \boldsymbol{\zeta}_{ik} \tag{11}$$

$$\boldsymbol{\nu}_k = \boldsymbol{\nu}^c + \boldsymbol{\Lambda}^c\boldsymbol{\eta}_k^c + \boldsymbol{\epsilon}_k^c \tag{12}$$

$$\boldsymbol{\eta}_k^c = \boldsymbol{\alpha}^c + \boldsymbol{B}^c\boldsymbol{\eta}_k^c + \boldsymbol{\zeta}_k^c \tag{13}$$

Model Summary

- So far, we have seen that traditional psychometric models involve linear relationships between the latent variables $\boldsymbol{\eta}$ and (functions of) the observed variables $\boldsymbol{y}$. And in SEM, we can have linear relationships between different elements of $\boldsymbol{\eta}$.
- We have described them as models of raw data, though they are also models of covariance matrices. It can be difficult to build intuition when we begin with covariance matrices.

Bayesian Introduction

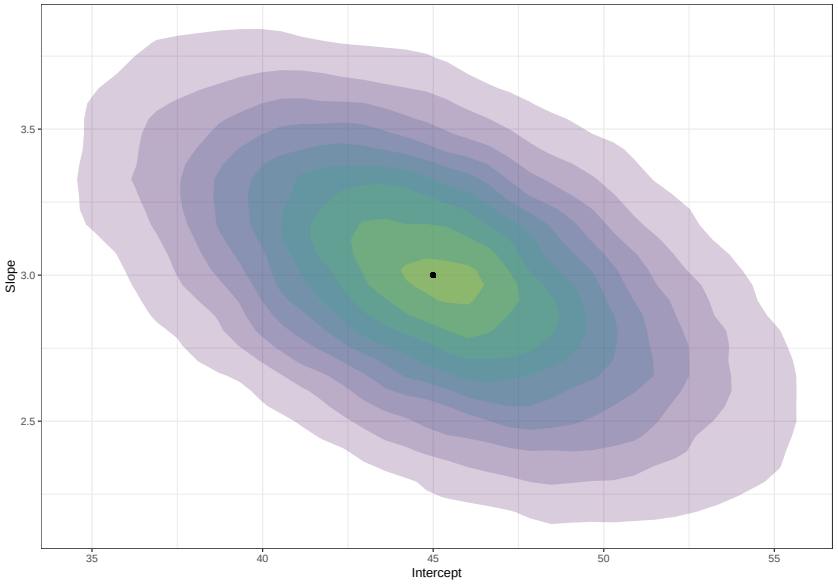
Model Estimation

- Traditionally, models are estimated via Maximum Likelihood. Idea:
  - The model defines a *likelihood function*. Inputs to the function are data and model parameters. Output is a single number ("the likelihood").
  - Each set of parameter values produce a single number as output.
  - We seek the parameter values that output the largest number, given our data.

Model Estimation

- How does this happen? Think about climbing a mountain.
  - Start somewhere on the mountain (start with some parameter values).
  - Decide which way is up.
  - Take a step in the upward direction (refine your parameter values).
  - Continue these steps until you reach the peak.
- The mountain is like a 2-parameter model. For models with more parameters, we need more than 3 dimensions so have no visuals!

Model Estimation



Bayesian Estimation

- The same model likelihood is involved in Bayesian estimation. But now we additionally include our prior expectations/beliefs about model parameters.
- These expectations are encoded as *prior distributions*. If you come from maximum likelihood estimation, we could view priors as a generalization of maximum likelihood:
  - Maximum likelihood: No/flat prior beliefs, anything can happen.
  - Bayesian: Prior beliefs could be flat, but they could also be informative.

Bayesian Estimation

- "Prior expectations sound subjective, and I don't want a subjective statistical analysis."
  - If it involves real data, some subjective decisions are already being made.
  - You might not know what parameter values to expect, but you probably know what you *do not* expect. Example: I developed a scale and do not expect my items to be negatively correlated.
  - Related to taking responsibility for one's models and analyses.

Identification

- Latent variables have no inherent location or scale.
  - It is customary to fix each latent variable's mean to 0 and variance to 1. (or, fix a loading to 1 instead of the variance.)
  - These *identification constraints* can influence our prior beliefs about parameter values. We will keep this in mind as we work through examples.
  - Sometimes, the identification constraints change the stated prior distribution. See Merkle, Ariyo, Winter, & Garnier-Villarreal (2023) at this hyperlink.

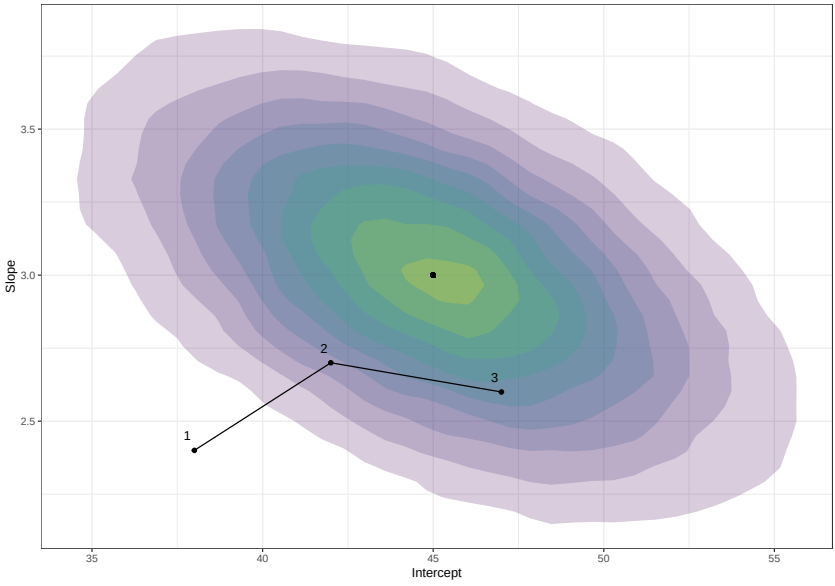
Bayesian Estimation

- Beyond prior distributions, Bayesian model estimation procedures usually differ from Maximum Likelihood:
  - Maximum likelihood is seeking the top of the mountain (of the likelihood)
  - Bayesian estimation typically surveys the full mountain, as opposed to only finding the top of the mountain. This is accomplished via Markov chain Monte Carlo.

MCMC

- Markov chain Monte Carlo: Draw samples of parameters from the posterior distribution.
  - Parameter values that are more likely will tend to be drawn more often.
  - If we draw many samples, we can produce accurate summaries of the posterior distribution.
  - But we need *many* samples, and this can use lots of computer memory!

MCMC



MCMC

- Specific flavors of MCMC include Gibbs sampling, Metropolis Hastings sampling, and Hamiltonian Monte Carlo.
- These can differ in speed, efficiency, and flexibility.
- We will discuss these in detail later.

Summary of Part 1

- Traditional psychometric models can be written in many ways, but they revolve around a linear relationship between the latent variables and the observed variables.
- Whereas maximum likelihood estimation involves the model likelihood function, Bayesian estimation involves the posterior distribution. This combines the likelihood function with prior beliefs about model parameters.
- When we estimate a Bayesian model, we often wish to learn about the full posterior distribution as opposed to only the peak.

Part 2: Practical Applications

Steps

- General steps that we will use in our Bayesian applications:
  - Set prior distributions, making use of prior predictive checks.
  - Estimate model via MCMC.
  - Do posterior predictive checks and other model criticisms.
  - Summarize key results.

Data

- For our applications, we use responses of 565 Austrian students to 7 mathematics items from PISA 2009.
- Conveniently included in the *sirt* package:

```
data("data.pisaMath", package = "sirt")

dat <- data.pisaMath$data
```

Data

- Obtaining response patterns

```
patts <- with(dat, paste0(M192Q01, M406Q01, M423Q01, M496Q01, M564Q01,
                          M571Q01, M603Q01))
```

Data

```
table(patts)
```

```
## patts
## 0000000 0000001 0000010 0000011 0000100 0000101 0000110 0000111 0001000 0001001 0001010
##      24       7       5       1       7       1       7       2       6       3       1
## 0001011 0001100 0001110 0001111 0010000 0010001 0010010 0010011 0010100 0010101 0010110
##       2       3       3       3      22      12       8       4      11       6       6
## 0010111 0011000 0011001 0011010 0011011 0011100 0011101 0011110 0011111 0100000 0100010
##       1      12      12       7       9       9       1       6      12       1       2
## 0100011 0100100 0100101 0100110 0100111 0101000 0101010 0101011 0101101 0101110 0101111
##       1       4       2       1       5       1       1       3       1       4       2
## 0110000 0110001 0110010 0110011 0110100 0110101 0110110 0110111 0111000 0111001 0111010
##       2       3       2       5       7       1       5       3       4       1       4
## 0111011 0111100 0111101 0111110 0111111 1000000 1000001 1000011 1000101 1000111 1001010
##       8       3       2       3      13       1       2       2       3       2       1
## 1001011 1001100 1001110 1001111 1010000 1010001 1010010 1010011 1010100 1010101 1010110
##       2       1       1       4       7       7       2       8       5       4       4
## 1010111 1011000 1011001 1011010 1011011 1011100 1011101 1011110 1011111 1100001 1100010
##       6       6       6       2       9       2       1       4      11       1       1
## 1100011 1100100 1100111 1101000 1101010 1101011 1101100 1101101 1101111 1110000 1110001
##       1       1       2       1       2       2       1       1      12       1       4
## 1110010 1110011 1110100 1110101 1110110 1110111 1111000 1111001 1111010 1111011 1111100
##       2       2       3       3       1       9       3       8       5      13       4
## 1111101 1111110 1111111
##      12       9      51
```

IRT Illustration

IRT

- To fit the data, we use an item response model that is specifically developed for binary responses.
- In the context of SEM, we might call it "item factor analysis". In the context of IRT, we might call it "graded response model with probit link". Or maybe "graded response model with normal ogive"?
- In short, the model predicts the z-score associated with the probability of being correct. The z-scores can be negative or positive, so it is no problem to have model predictions below 0 or above 1.

Model

- Description of model parameters:
  - Intercept (in SEM, threshold): Larger numbers mean you are less likely to be correct. So sometimes called *item difficulty*.
  - Slope: How do person parameters influence chance (z-score) of being correct?
  - Residual standard deviations: Fixed to 1, because variance of 0/1 data is determined by the mean.
  - Latent variable (factor) mean fixed to 0, latent variable variance fixed to 1.

Model

- Note: IRT modelers will often use an equivalent difficulty/discrimination parameterization, instead of slope/intercept! It is possible to obtain one set of parameters from the other.
  - Slope/intercept: $-\beta_{0j} + \beta_{1j}\theta_i$
  - Discrimination/difficulty: $\alpha_{1j}(\theta_i - \delta_{0j})$
- Also note: If you fit this model from an SEM perspective, the intercepts become *thresholds*. Intercepts and thresholds could be distinguished from each other in certain multi-group models, but not in the models we will examine.

Model

- Also note: IRT modelers will often predict the log-odds of being correct, instead of the z-score of being correct!
  - This is the difference between the logit link function (log odds) and the probit link function (z-score).
  - So our model could be called a two-parameter probit, vs a two-parameter logistic.
  - Probit parameter estimates can be converted to logit estimates, and vice versa. (see, e.g., McDonald, 1999)

Priors

- When setting priors, it helps to remember that we are predicting z-scores. It would be unusual to observe a number outside of $(-3, 3)$. So some initial prior distributions could be:
  - Intercepts: Normal(0, 1) $\leftarrow$ second number is SD
  - Slopes (loadings): Normal(1, .5). We expect all the items to be positively related to θ (i.e., no reverse coding), so we place most prior density on positive slope values.

Priors

- We specify the prior distributions in blavaan, where intercepts are in tau (thresholds).

```
## blavaan defaults:
dpriors()
```

```
##                nu             alpha            lambda              beta             theta
##    "normal(0,32)"    "normal(0,10)"    "normal(0,10)"    "normal(0,10)" "gamma(1,.5)[sd]"
##               psi               rho             ibpsi               tau
## "gamma(1,.5)[sd]"       "beta(1,1)" "wishart(3,iden)"   "normal(0,1.5)"
```

```
## replacing the defaults for thresholds and loadings:
mypriors <- dpriors(tau = "normal(0, 1.5)", lambda = "normal(1, .5)")
```

Priors

- We generate parameters from the priors, which is helpful for assessing whether our priors are reasonable. Note `prisamp = TRUE`:

```
## specify the model:
m2 <- ' f1 =~ M192Q01 + M406Q01 + M423Q01 + M496Q01 + M564Q01 + M571Q01 + M603Q01 '

## drawing prior samples (100 for each of three chains):
m2pri <- bcfa(m2, data = dat, burnin = 100, sample = 100, std.lv = TRUE, prisamp = TRUE,
              dp = mypriors, ordered = TRUE)

##
## SAMPLING FOR MODEL 'stanmarg' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001936 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 19.36 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: There aren't enough warmup iterations to fit the
## Chain 1:          three stages of adaptation as currently configured.
## Chain 1:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 1:          the given number of warmup iterations:
## Chain 1:            init_buffer = 15
## Chain 1:            adapt_window = 75
```

Priors

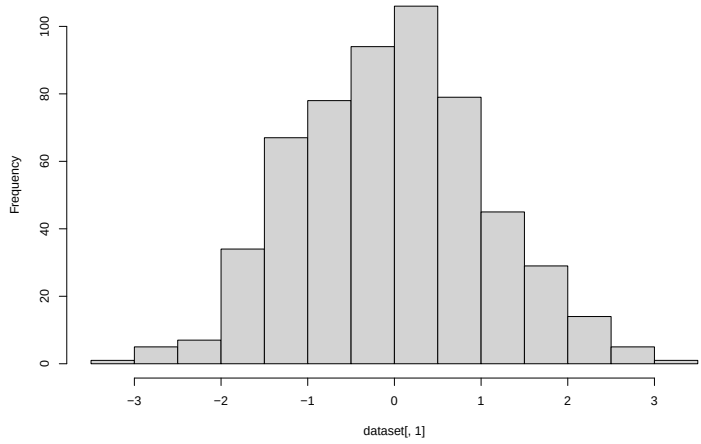
- Now we can generate data from the priors. The `type = "link"` argument says that we want to generate z-scores associated with the probability of being correct, as opposed to the original 0/1 data:

```
pridat <- sampleData(m2pri, simplify = TRUE, type = "link")
```

Priors

- Histogram of the first variable from one generated dataset:

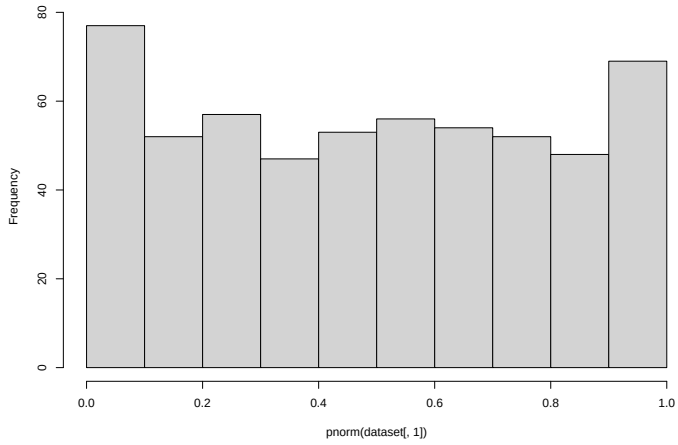
```
dataset <- pridat[[ 1 ]]
hist(dataset[, 1], main = "")
```



Priors

- Translated to probabilities of being correct:

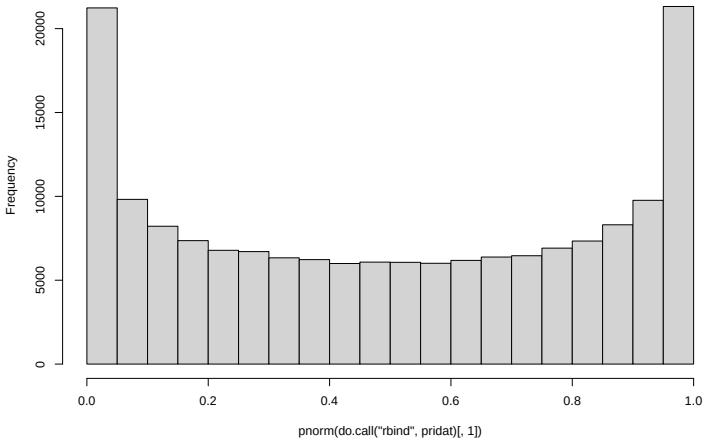
```
hist(pnorm( dataset[, 1] ), main = "")
```



Priors

- The previous histogram produced data where people had extreme probabilities of being correct. Does it hold across all the generated datasets?

```
hist(pnorm( do.call("rbind", pridat)[, 1] ), main = "")
```



Priors

- If the items are good, then we would expect test-takers to have a probability of being correct that is not too close to 0 or 1. So we might revise our priors to produce less-extreme probabilities.
  - Intercept: z-score associated with probability that the average test-taker gets the item correct. Say that the probability ranges from around .2 to .8, which are z-scores of $\pm .84$. Normal(0, .28)
  - Slope: For two test-takers whose proficiencies differ by 1 SD, what is the expected increase in probability correct? On probit scale, an increase of 1 could go from 16% chance to 50% chance, which is large. So Normal(.4, .2)

Priors

- The discussion on the previous slide leads to

```
mypriors <- dpriors(tau = "normal(0, .28)", lambda = "normal(.4, .2)")
```

Posteriors

- We claim that our prior distributions reflect general expectations, but are only mildly informative.
- We can do a sensitivity analysis to explore how our priors influence results.
- So we fit two models, one with our informative priors and one without.

Posteriors

```
m2fit <- bcfa(m2, data = dat, std.lv = TRUE, ordered = TRUE, dp = mypriors)
```

```
##
## SAMPLING FOR MODEL 'stanmarg' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001617 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 16.17 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 1500 [  0%]  (Warmup)
## Chain 1: Iteration:  150 / 1500 [ 10%]  (Warmup)
## Chain 1: Iteration:  300 / 1500 [ 20%]  (Warmup)
## Chain 1: Iteration:  450 / 1500 [ 30%]  (Warmup)
## Chain 1: Iteration:  501 / 1500 [ 33%]  (Sampling)
## Chain 1: Iteration:  650 / 1500 [ 43%]  (Sampling)
## Chain 1: Iteration:  800 / 1500 [ 53%]  (Sampling)
## Chain 1: Iteration:  950 / 1500 [ 63%]  (Sampling)
## Chain 1: Iteration: 1100 / 1500 [ 73%]  (Sampling)
## Chain 1: Iteration: 1250 / 1500 [ 83%]  (Sampling)
## Chain 1: Iteration: 1400 / 1500 [ 93%]  (Sampling)
## Chain 1: Iteration: 1500 / 1500 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 51.225 seconds (Warm-up)
## Chain 1:                51.968 seconds (Sampling)
```

Posteriors

```
m2nifit <- bcfa(m2, data = dat, std.lv = TRUE, ordered = TRUE)
```

```
##
## SAMPLING FOR MODEL 'stanmarg' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001711 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 17.11 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 1500 [  0%]  (Warmup)
## Chain 1: Iteration:  150 / 1500 [ 10%]  (Warmup)
## Chain 1: Iteration:  300 / 1500 [ 20%]  (Warmup)
## Chain 1: Iteration:  450 / 1500 [ 30%]  (Warmup)
## Chain 1: Iteration:  501 / 1500 [ 33%]  (Sampling)
## Chain 1: Iteration:  650 / 1500 [ 43%]  (Sampling)
## Chain 1: Iteration:  800 / 1500 [ 53%]  (Sampling)
## Chain 1: Iteration:  950 / 1500 [ 63%]  (Sampling)
## Chain 1: Iteration: 1100 / 1500 [ 73%]  (Sampling)
## Chain 1: Iteration: 1250 / 1500 [ 83%]  (Sampling)
## Chain 1: Iteration: 1400 / 1500 [ 93%]  (Sampling)
## Chain 1: Iteration: 1500 / 1500 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 56.242 seconds (Warm-up)
## Chain 1:                50.244 seconds (Sampling)
```

Posteriors I

```
summary(m2fit)

## blavaan 0.5.8 ended normally after 1000 iterations
##
##   Estimator                                      BAYES
##   Optimization method                             MCMC
##   Number of model parameters                        14
##
##   Number of observations                           565
##   Number of missing patterns                         1
##
##   Statistic                                 MargLogLik         PPP
##   Value                                      -2520.821       0.269
##
## Parameter Estimates:
##
##   Parameterization                               Theta
##
## Latent Variables:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##   f1 =~
##     M192Q01           0.774    0.094    0.602    0.959    1.006 normal(.4, .2)
##     M406Q01           0.812    0.095    0.626    1.003    1.000 normal(.4, .2)
##     M423Q01           0.328    0.072    0.189    0.475    1.001 normal(.4, .2)
##     M496Q01           0.660    0.087    0.490    0.839    1.005 normal(.4, .2)
##     M564Q01           0.486    0.077    0.336    0.645    1.003 normal(.4, .2)
##     M571Q01           0.718    0.092    0.544    0.905    1.000 normal(.4, .2)
##     M603Q01           0.700    0.091    0.529    0.890    1.007 normal(.4, .2)
##
## Intercepts:
```

Posteriors II

```
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##    .M192Q01           0.000
##    .M406Q01           0.000
##    .M423Q01           0.000
##    .M496Q01           0.000
##    .M564Q01           0.000
##    .M571Q01           0.000
##    .M603Q01           0.000
##     f1                0.000
##
## Thresholds:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##     M192Q01|t1        0.139    0.064    0.011    0.261    1.001 normal(0, .28)
##     M406Q01|t1        0.189    0.066    0.055    0.318    1.001 normal(0, .28)
##     M423Q01|t1       -0.648    0.059   -0.763   -0.534    1.003 normal(0, .28)
##     M496Q01|t1       -0.157    0.061   -0.272   -0.035    1.000 normal(0, .28)
##     M564Q01|t1       -0.039    0.058   -0.151    0.075    1.000 normal(0, .28)
##     M571Q01|t1       -0.137    0.062   -0.259   -0.013    1.000 normal(0, .28)
##     M603Q01|t1       -0.164    0.061   -0.284   -0.043    1.000 normal(0, .28)
##
## Variances:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##    .M192Q01           1.000
##    .M406Q01           1.000
##    .M423Q01           1.000
##    .M496Q01           1.000
##    .M564Q01           1.000
##    .M571Q01           1.000
##    .M603Q01           1.000
##     f1                1.000
```

Posteriors III

```
##
## Scales y*:
##                  Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##     M192Q01         0.791
##     M406Q01         0.776
##     M423Q01         0.950
##     M496Q01         0.835
##     M564Q01         0.899
##     M571Q01         0.812
##     M603Q01         0.819
```

Sensitivity

- Comparing informative priors to default priors, the informative priors appear to be no worse by simple cross-validation metrics:

```
fitMeasures(m2fit)
```

```
##       npar       logl        ppp        bic        dic      p_dic       waic     p_waic
##     14.000  -2493.290      0.269   5075.270   5010.751     12.086   5010.399     11.703
##    se_waic      looic      p_loo     se_loo margloglik
##     43.441   5010.436     11.721     43.441  -2520.821
```

```
fitMeasures(m2nifit)
```

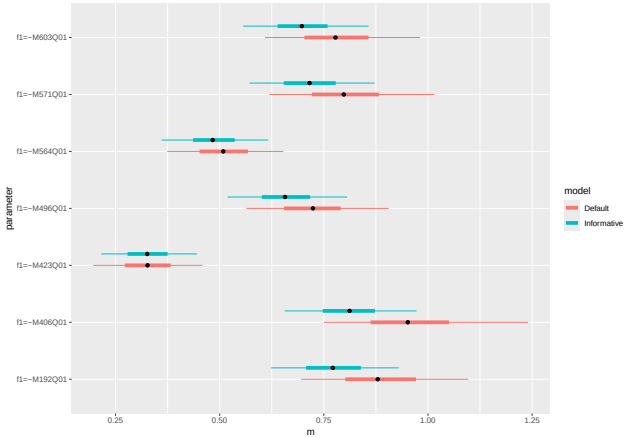
```
##       npar       logl        ppp        bic        dic      p_dic       waic     p_waic
##     14.000  -2491.699      0.283   5072.089   5012.126     14.364   5012.680     14.854
##    se_waic      looic      p_loo     se_loo margloglik
##     47.299   5012.722     14.876     47.300  -2545.723
```

Sensitivity I

- Comparing loadings under the two models:

```
### related to code at https://github.com/stan-dev/bayesplot/issues/232
combined <- rbind(plot(m2fit, 1:7, 'intervals_data', showplot = FALSE),
                  plot(m2nifit, 1:7, 'intervals_data', showplot = FALSE))
combined$model <- rep(c("Informative", "Default"), each = nrow(combined)/2)

pos <- position_nudge(y = ifelse(combined$model == "Default", 0, 0.2))

ggplot(combined, aes(x = m, y = parameter, color = model)) +
  geom_linerange(aes(xmin = l, xmax = h), position = pos, linewidth = 2)+
  geom_linerange(aes(xmin = ll, xmax = hh), position = pos)+
  geom_point(position = pos, color="black")
```

Sensitivity II



Sensitivity

- The graph shows, under informative priors, the loadings are shrunk toward zero.
- I am not generally concerned about this because loadings are difficult to estimate (like estimating an interaction between person and item, as opposed to a main effect). It is doubtful that shrinkage will hurt the model's generalizability.
- The WAIC and LOO metrics also support the idea that informative priors could provide better generalization. (If you don't know these, they have similar goals to AIC or BIC.)
- But you might also consider the purpose of the model.

Model Checking

- The posterior predictive p-value indicates reasonable model fit: generally, values closer to 0.5 than to 0 are good. We can dig deeper with customized posterior predictive checks via `ppmc()`.
- There is a good deal of flexibility here because you can write custom R functions involving a blavaan object.
- Example: Bonifay & Depaoli (2021) describe use of the item-total correlation for model checking. We examine the empirical correlation between each item and the sum of remaining items, and compare to the posterior distribution of the same correlation.

Model Checking

- Define a function that computes item-total correlations, then send to `ppmc()`:

```
it_tot <- function(fit) {
  tmpdata <- fit@Data@X[[1]]
  sapply(1:ncol(tmpdata),
         function(i) cor(tmpdata[,i], rowSums(tmpdata[,-i])))
}

itt2 <- ppmc(m2fit, discFUN = it_tot)
```

Model Checking

- These ppps indicate that the observed item-total correlations fall inside the posterior predictive distributions. Some columns indicate no variability because the observed item-total correlation is a function of data only (as opposed to function of data and model parameters).

```
summary(itt2)

##
## Posterior summary statistics and highest posterior density (HPD) 95% credible intervals
##  for the posterior distribution of realized discrepancy-function values based on observed data,
##  along with posterior predictive p values to test hypotheses in either direction:
##
##
##     EAP Median   MAP SD lower upper PPP_sim_GreaterThan_obs PPP_sim_LessThan_obs
## 1 0.415  0.415 0.415  0 0.415 0.415                   0.129                0.871
## 2 0.424  0.424 0.423  0 0.424 0.424                   0.142                0.858
## 3 0.176  0.176 0.176  0 0.176 0.176                   0.526                0.474
## 4 0.368  0.368 0.368  0 0.368 0.368                   0.239                0.761
## 5 0.264  0.264 0.264  0 0.264 0.264                   0.517                0.483
## 6 0.386  0.386 0.386  0 0.386 0.386                   0.203                0.797
## 7 0.371  0.371 0.371  0 0.371 0.371                   0.291                0.709
```

Summary

- We estimated our Bayesian IRT model as a factor analysis of discrete data.
  - We could call it either item factor analysis or item response model: estimation methods often differ for frequentist models, but they are similar in Bayesian modeling.
  - We generated prior predictive data to help consider whether our priors were reasonable.
  - We saw some model checks and comparisons that are especially flexible in *blavaan*, because we can use the R universe to define functions for posterior checks.

Explanatory IRT Extension

Explanatory IRT

- We just considered a traditional Bayesian IRT model. That model has many uses, including item selection, adaptive testing, scoring/estimating person parameters.
- But the previous IRT model might be unsatisfying if we want to understand *why* people responded in the way that they did:
  - Why is item 1 more difficult than item 2?
  - Why is person 1 more proficient than person 2?

Explanatory IRT

- We can start to address these questions by including extra covariates in the model. Then we might call our model an *explanatory* item response model (see De Boeck & Wilson, 2004).
- Bayesian SEM provides a good deal of flexibility for estimating these models.

Explanatory IRT

- Some possibilities:
  - Use covariates to predict a person's proficiency.
  - Decompose item difficulties into effects associated with specific item attributes.
  - Person-by-item interactions, getting at differential item functioning.

Explanatory IRT

- Here, we expand on our previous model by predicting proficiency via two person covariates: female (0/1) and SES (centered/scaled).
- The model provides information about how the two covariates are predictive of a person's proficiency across the 7 items.
- Our outcome variable (proficiency) is unobserved. Bayesian methods can help us characterize uncertainty in the estimated relationships between observed covariates and unobserved outcome.

Model

- From a *blavaan* point of view, including covariates is a simple addition to the previous model syntax.

```
m3 <- ' f1 =~ M192Q01 + M406Q01 + M423Q01 + M496Q01 + M564Q01 + M571Q01 + M603Q01
        f1 ~ female + hisei + female:hisei '
```

Model

- We now use `bsem()` because the model includes regressions involving latent variables.
- Also, `fixed.x = TRUE` treats our covariates as fixed (as in traditional regression).
- Also, `save.lvs = TRUE` samples the latent variables and will facilitate model checking.

Model

```
m3fit <- bsem(m3, data = dat, std.lv = TRUE, ordered = TRUE, dp = mypriors,
              fixed.x = TRUE, save.lvs = TRUE)
```

```
##
## SAMPLING FOR MODEL 'stanmarg' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.002722 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 27.22 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 1500 [  0%]  (Warmup)
## Chain 1: Iteration:  150 / 1500 [ 10%]  (Warmup)
## Chain 1: Iteration:  300 / 1500 [ 20%]  (Warmup)
## Chain 1: Iteration:  450 / 1500 [ 30%]  (Warmup)
## Chain 1: Iteration:  501 / 1500 [ 33%]  (Sampling)
## Chain 1: Iteration:  650 / 1500 [ 43%]  (Sampling)
## Chain 1: Iteration:  800 / 1500 [ 53%]  (Sampling)
## Chain 1: Iteration:  950 / 1500 [ 63%]  (Sampling)
## Chain 1: Iteration: 1100 / 1500 [ 73%]  (Sampling)
## Chain 1: Iteration: 1250 / 1500 [ 83%]  (Sampling)
## Chain 1: Iteration: 1400 / 1500 [ 93%]  (Sampling)
## Chain 1: Iteration: 1500 / 1500 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 74.756 seconds (Warm-up)
```

Results I

```
summary(m3fit)

## blavaan 0.5.8 ended normally after 1000 iterations
##
##   Estimator                                      BAYES
##   Optimization method                             MCMC
##   Number of model parameters                        17
##
##   Number of observations                           565
##   Number of missing patterns                         1
##
##   Statistic                                 MargLogLik         PPP
##   Value                                      -2512.006       0.054
##
## Parameter Estimates:
##
##   Parameterization                               Theta
##
## Latent Variables:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##   f1 =~
##     M192Q01           0.760    0.092    0.593    0.947    1.000 normal(.4, .2)
##     M406Q01           0.747    0.093    0.575    0.929    1.000 normal(.4, .2)
##     M423Q01           0.308    0.068    0.179    0.446    1.000 normal(.4, .2)
##     M496Q01           0.630    0.084    0.477    0.802    1.002 normal(.4, .2)
##     M564Q01           0.431    0.071    0.292    0.578    1.000 normal(.4, .2)
##     M571Q01           0.693    0.091    0.529    0.882    0.999 normal(.4, .2)
##     M603Q01           0.650    0.083    0.496    0.820    1.001 normal(.4, .2)
##
## Regressions:
```

Results II

```
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##   f1 ~
##     female          -0.203    0.108   -0.424   -0.001    1.001   normal(0,10)
##     hisei            0.442    0.081    0.288    0.601    0.999   normal(0,10)
##     female:hisei    -0.202    0.111   -0.420    0.011    1.000   normal(0,10)
##
## Intercepts:
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##    .M192Q01          0.000
##    .M406Q01          0.000
##    .M423Q01          0.000
##    .M496Q01          0.000
##    .M564Q01          0.000
##    .M571Q01          0.000
##    .M603Q01          0.000
##    .f1               0.000
##
## Thresholds:
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##     M192Q01|t1       0.085    0.076   -0.064    0.228    1.001 normal(0, .28)
##     M406Q01|t1       0.137    0.074   -0.009    0.284    1.000 normal(0, .28)
##     M423Q01|t1      -0.670    0.063   -0.796   -0.553    1.001 normal(0, .28)
##     M496Q01|t1      -0.200    0.070   -0.335   -0.069    1.000 normal(0, .28)
##     M564Q01|t1      -0.069    0.060   -0.185    0.047    1.001 normal(0, .28)
##     M571Q01|t1      -0.188    0.071   -0.332   -0.047    1.002 normal(0, .28)
##     M603Q01|t1      -0.208    0.071   -0.350   -0.069    1.000 normal(0, .28)
##
## Variances:
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##    .M192Q01          1.000
```

Results III

```
##    .M406Q01          1.000
##    .M423Q01          1.000
##    .M496Q01          1.000
##    .M564Q01          1.000
##    .M571Q01          1.000
##    .M603Q01          1.000
##    .f1               1.000
##
## Scales y*:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##     M192Q01           0.776
##     M406Q01           0.781
##     M423Q01           0.950
##     M496Q01           0.829
##     M564Q01           0.908
##     M571Q01           0.803
##     M603Q01           0.821
```

Results

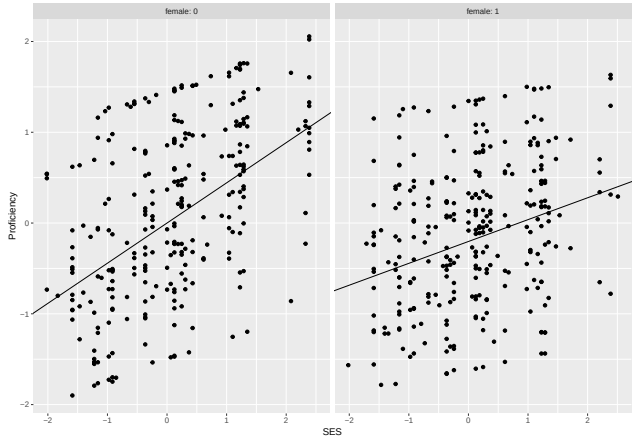
- The regression estimates imply a negative association between female and proficiency, and a positive association between SES and proficiency.
- The posterior interval for the interaction overlaps with 0 but is mostly negative.
- We can view the model as estimating two regression lines, one for female and one for non-female:
  - Non-female: intercept 0, slope 0.442
  - Female: intercept -0.203, slope $(0.442 + (-0.202))$
  - (these are posterior means, there is also uncertainty in the estimates!)

Results I

- Graph of posterior mean regression lines, and posterior mean latent variables:

```
lvmeans <- blavInspect(m3fit, 'lvmeans')
dat$f1pred <- lvmeans[, 1]

regwts <- coef(m3fit)[grep("^f1~", names(coef(m3fit)))]
regdf <- cbind.data.frame(female = c(0, 1), int = c(0, regwts[1]), slp = c(regwts[2], sum(regwts[2:3])))

ggplot(dat, aes(x = hisei, y = f1pred)) + geom_point() + geom_abline(data = regdf, aes(slope = slp, intercept = int)) +
  facet_wrap( ~ female, labeller = label_both) + xlab("SES") + ylab("Proficiency")
```

Results II

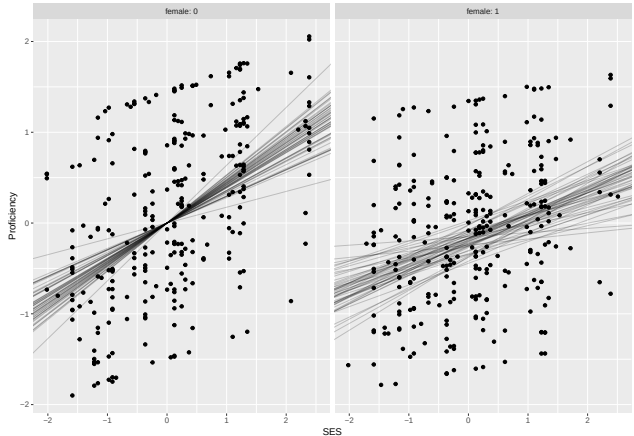


Results I

- Same graph, but show uncertainty in the regression line. We do this by drawing the lines of 50 posterior samples:

```
p <- ggplot(dat, aes(x = hisei, y = f1pred)) + geom_point() + facet_wrap( ~ female, labeller = label_both) +
  xlab("SES") + ylab("Proficiency")

samps <- do.call("rbind", blavInspect(m3fit, 'mcmc'))
ndraws <- 50
regdf <- cbind.data.frame(female = rep(c(0, 1), ndraws), int = rep(0, ndraws * 2), slp = rep(0, ndraws * 2))
draws <- sample(1:nrow(samps), ndraws)
for (i in 1:length(draws)) {
  regwts <- samps[draws[i], grep("^f1~", colnames(samps))]
  regdf$int[i * 2] <- regwts[1]
  regdf$slp[(i - 1)*2 + 1:2] <- c(regwts[2], sum(regwts[2:3]))
}

p + geom_abline(data = regdf, aes(slope = slp, intercept = int), alpha=.2)
```

Results II



Results

- The previous graph shows an odd characteristic: the line for non-female is constrained to go through (0,0)
  - This is the intercept for non-females.
  - This intercept is fixed to 0, because it identifies the latent variable.
  - This does not cause as many problems as one may expect, because the 0 point on the latent variable is arbitrary.
  - But if we “center” female, we can obtain visualizations that are less unusual.

Model

- Centering and re-estimating the model:

```
dat$female <- dat$female - .5

m4fit <- bsem(m3, data = dat, std.lv = TRUE, ordered = TRUE, dp = mypriors,
              fixed.x = TRUE, save.lvs = TRUE)
```

```
##
## SAMPLING FOR MODEL 'stanmarg' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.002138 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 21.38 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 1500 [  0%]  (Warmup)
## Chain 1: Iteration:  150 / 1500 [ 10%]  (Warmup)
## Chain 1: Iteration:  300 / 1500 [ 20%]  (Warmup)
## Chain 1: Iteration:  450 / 1500 [ 30%]  (Warmup)
## Chain 1: Iteration:  501 / 1500 [ 33%]  (Sampling)
## Chain 1: Iteration:  650 / 1500 [ 43%]  (Sampling)
## Chain 1: Iteration:  800 / 1500 [ 53%]  (Sampling)
## Chain 1: Iteration:  950 / 1500 [ 63%]  (Sampling)
## Chain 1: Iteration: 1100 / 1500 [ 73%]  (Sampling)
## Chain 1: Iteration: 1250 / 1500 [ 83%]  (Sampling)
```

Results I

```
summary(m4fit)

## blavaan 0.5.8 ended normally after 1000 iterations
##
##   Estimator                                      BAYES
##   Optimization method                             MCMC
##   Number of model parameters                        17
##
##   Number of observations                           565
##   Number of missing patterns                         1
##
##   Statistic                                 MargLogLik         PPP
##   Value                                      -2511.574       0.065
##
## Parameter Estimates:
##
##   Parameterization                               Theta
##
## Latent Variables:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##   f1 =~
##     M192Q01           0.766    0.092    0.594    0.951    1.001 normal(.4, .2)
##     M406Q01           0.749    0.089    0.580    0.934    1.000 normal(.4, .2)
##     M423Q01           0.315    0.067    0.187    0.449    1.000 normal(.4, .2)
##     M496Q01           0.630    0.079    0.473    0.787    1.001 normal(.4, .2)
##     M564Q01           0.428    0.071    0.293    0.568    1.001 normal(.4, .2)
##     M571Q01           0.688    0.089    0.518    0.868    1.004 normal(.4, .2)
##     M603Q01           0.649    0.081    0.497    0.815    1.000 normal(.4, .2)
##
## Regressions:
```

Results II

```
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##   f1 ~
##     female          -0.238    0.116   -0.467   -0.010    1.001   normal(0,10)
##     hisei            0.340    0.056    0.230    0.446    0.999   normal(0,10)
##     female:hisei    -0.198    0.113   -0.425    0.019    1.002   normal(0,10)
##
## Intercepts:
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##    .M192Q01          0.000
##    .M406Q01          0.000
##    .M423Q01          0.000
##    .M496Q01          0.000
##    .M564Q01          0.000
##    .M571Q01          0.000
##    .M603Q01          0.000
##    .f1               0.000
##
## Thresholds:
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##     M192Q01|t1       0.153    0.067    0.020    0.287    0.999 normal(0, .28)
##     M406Q01|t1       0.205    0.066    0.076    0.338    1.000 normal(0, .28)
##     M423Q01|t1      -0.644    0.059   -0.761   -0.525    1.002 normal(0, .28)
##     M496Q01|t1      -0.145    0.061   -0.266   -0.025    1.001 normal(0, .28)
##     M564Q01|t1      -0.031    0.056   -0.138    0.080    1.002 normal(0, .28)
##     M571Q01|t1      -0.125    0.062   -0.244   -0.003    1.001 normal(0, .28)
##     M603Q01|t1      -0.151    0.062   -0.275   -0.032    1.000 normal(0, .28)
##
## Variances:
##                   Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##    .M192Q01          1.000
```

Results III

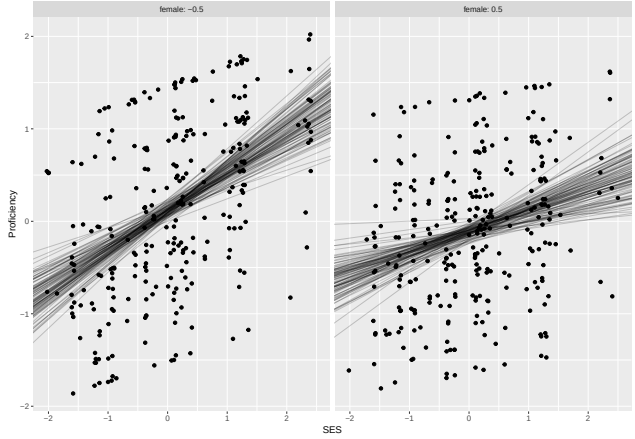
```
##    .M406Q01          1.000
##    .M423Q01          1.000
##    .M496Q01          1.000
##    .M564Q01          1.000
##    .M571Q01          1.000
##    .M603Q01          1.000
##    .f1               1.000
##
## Scales y*:
##                    Estimate  Post.SD pi.lower pi.upper     Rhat    Prior
##     M192Q01           0.773
##     M406Q01           0.780
##     M423Q01           0.947
##     M496Q01           0.829
##     M564Q01           0.909
##     M571Q01           0.805
##     M603Q01           0.821
```

Results I

- Re-draw the plot:

```
lvmeans <- blavInspect(m4fit, 'lvmeans')
dat$f1pred <- lvmeans[, 1]

p <- ggplot(dat, aes(x = hisei, y = f1pred)) + geom_jitter() + facet_wrap( ~ female, labeller = label_both) +
  xlab("SES") + ylab("Proficiency")

samps <- do.call("rbind", blavInspect(m4fit, 'mcmc'))
ndraws <- 100
regdf <- cbind.data.frame(female = rep(c(-.5, .5), ndraws), int = rep(0, ndraws * 2), slp = rep(0, ndraws * 2))
draws <- sample(1:nrow(samps), ndraws)
for (i in 1:length(draws)) {
  regwts <- samps[draws[i], grep("^f1~", colnames(samps))]
  regdf$int[(i - 1)*2 + 1:2] <- regwts[1] * c(-.5, .5)
  regdf$slp[(i - 1)*2 + 1:2] <- regwts[2] + regwts[3] * c(-.5, .5)
}

p + geom_abline(data = regdf, aes(slope = slp, intercept = int), alpha=.2)
```

Results II



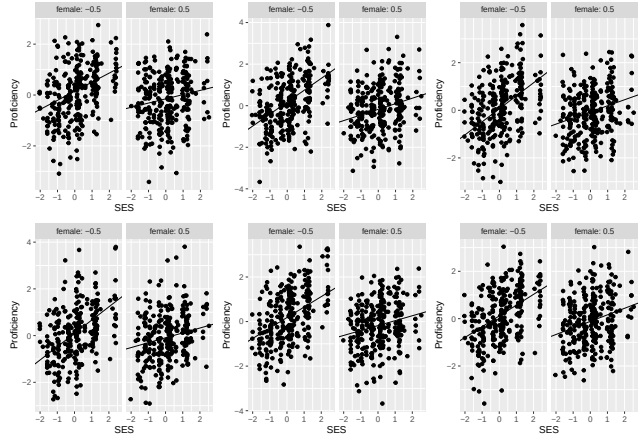
Results

- The previous graphs represent uncertainty in the model's regression parameters.
- But there is also uncertainty in the points in the y-axis direction (in the latent variables)! This uncertainty is more difficult to capture visually.
- One possibility: separate scatterplots for each posterior sample (which includes latent variables).

Results I

```
## each panel represents one posterior sample
library("patchwork")

lvs <- do.call("rbind", blavInspect(m4fit, 'lvs'))
ndraws <- 6
draws <- sample(1:nrow(samps), ndraws)

ps <- vector("list", length(draws))

for (i in 1:ndraws) {
  dat$f1pred <- lvs[draws[i], ]

  regwts <- samps[draws[i], grep("^f1~", colnames(samps))]
  regdf <- cbind.data.frame(female = c(-.5, .5), int = regwts[1] * c(-.5, .5), slp = regwts[2] + regwts[3] * c(-.5, .5))

  ps[[i]] <- ggplot(dat, aes(x = hisei, y = f1pred)) + geom_jitter() +
    geom_abline(data = regdf, aes(slope = slp, intercept = int)) +
    facet_wrap( ~ female, labeller = label_both) +
    xlab("SES") + ylab("Proficiency")
}

Reduce("+", ps)
```

Results II



Summary

- The *blavaan* framework facilitates extension of traditional IRT models to explanatory IRT models.
- The graphical capabilities of R help us find modeling problems/issues that may be difficult or impossible to find by staring at text output.
- In turn, the above attributes help us to best characterize the implications of the model for the observed data.

Part 3: Traditional MCMC

MCMC

- Specific MCMC methods include Gibbs sampling, Metropolis Hastings sampling, and Hamiltonian Monte Carlo.
- These can differ in speed, efficiency, and flexibility.
- We will now build some intuition about how the MCMC methods work, and see how we could use them for psychometric modeling.

History

- Gibbs sampling received much attention in the 80s and 90s. Near the end of the 90s, WinBUGS provided flexibility to the masses. (though WinBUGS had a variety of samplers, not just Gibbs!)
- Metropolis-Hastings sampling supplements Gibbs sampling, being especially useful for sampling from nonstandard models.
- Hamiltonian sampling is more recent, providing improved sampling efficiency.

Overview

- Rough idea of what they involve:
  - Gibbs: Instead of sampling all parameters at once, sample individual parameters conditional on the values of other parameters.
  - Metropolis-Hastings: If we don't know how to sample a particular parameter, sample from some easy distribution and accept the sample with a specific probability.
  - Hamiltonian: Use the model gradient to improve our sampling.

Overview

- Remember our simple, two-parameter mountain? We can visualize the MCMC methods in a similar way:

https://chi-feng.github.io/mcmc-demo/app.html

Outline of Part 3

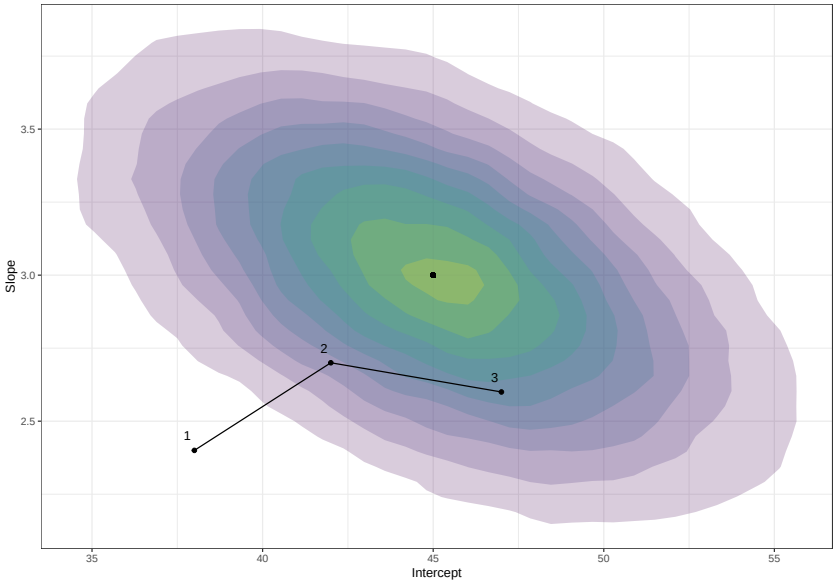
- The goal of this part is to understand how to construct traditional Gibbs samplers for psychometric models.
  - Use of conjugate prior distributions.
  - Start with regression results, build to SEM.
  - Consider how we can use lavaan to help us build samplers.

Gibbs Overview

Gibbs Sampling

- Most psychometric models have many parameters, including intercepts, loadings, variances, and covariances.
- Sampling from the posterior distribution involves repeatedly sampling all these parameters.
- The "Markov chain" part of MCMC implies that we sample new parameter values conditioned on the current parameter values.

Visualization



Visualization

- Referring to the previous slide:
  - A joint update would involve moving directly from point 1 to point 2, then from point 2 to point 3. For example, the first step moves from (38,2.4) to (42,2.7).
  - In Gibbs sampling, we could separately sample the intercept and slope. So, for the first step, we have an intermediate move from (38,2.4) to (38,2.7). Then a second intermediate move from (38,2.7) to (42,2.7).
  - This can simplify sampling because it allows us to deal with distributions of smaller dimension. It can also allow us to take advantage of conjugate prior distributions.

Conjugacy

- Conjugate prior distribution: A prior on a (sub)set of model parameters such that the posterior distribution is in the same family as the prior distribution.
  - This can allow us to analytically sample from some posterior distributions (using, e.g., `rnorm()` or other random number generators).
  - And Gibbs sampling allows us to work with conjugate prior distributions of individual parameters, as opposed to the full prior distribution of all parameters.

Conjugacy

- Conjugate prior distributions by parameter type
  - Intercept/loading/regression weights: Normal (multivariate normal for many parameters at once)
  - Variances: Inverse gamma (scaled-inverse χ^2)
  - Covariance matrices: Inverse Wishart

Summary

- So Gibbs sampling allows us to sample individual parameters sequentially.
- And we know conjugate priors of many individual parameters, which means we can analytically sample from those parameters' posteriors.
- Taken together, these ideas lead us to traditional Gibbs samplers for many statistical models that involve linear equations.

Regression

Regression

- Gibbs samplers for psychometric models can become very notationally heavy. This makes it difficult to understand the intuition behind the major results, and how the samplers work in practice.
- But many of the main results come from regression. So it is helpful to first look at Gibbs samplers for regression, then consider how they can be extended to psychometrics. Our discussion here is related to Gelman et al. (2013).

Model

- The usual regression model in matrix form:

$$\boldsymbol{y} \sim \mathrm{N}(\boldsymbol{X}\beta, \sigma^2\boldsymbol{I}), \tag{14}$$

- We are modeling all n observations at once, using vectors and matrices!
- The model covariance matrix, $\sigma^2\boldsymbol{I}$, represents the assumptions of independence and homogeneity.

Conditional Posteriors

- To start, let's assume flat priors on β and on $\log(\sigma)$. Then we can show that the conditional posterior distribution of β is

$$\beta \mid \sigma, \boldsymbol{y} \sim \mathrm{N}((\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y}, (\boldsymbol{X}'\boldsymbol{X})^{-1}\sigma^2). \tag{15}$$

  - This especially makes sense because the mean of the distribution is the usual least squares estimate of β.

Conditional Posteriors

- What about the conditional posterior of σ^2?

$$\sigma^2 \mid \boldsymbol{y}, \beta \sim \text{Scale-inv-}\chi^2 \left(n, \frac{1}{n}(\boldsymbol{y} - \boldsymbol{X}\beta)'(\boldsymbol{y} - \boldsymbol{X}\beta) \right) \tag{16}$$

  - The Scaled-Inverse-χ^2 is just a reparameterized Inverse-Gamma distribution.
  - We recognize the second parameter as being related to the mean squared error of the regression model.

Relaxed Assumptions

- As specified, the regression model has a diagonal covariance matrix $\sigma^2 \boldsymbol{I}$ with only one free parameter.
- Imagine that we relax this assumption so that our regression model is

$$\boldsymbol{y} \sim \mathrm{N}(\boldsymbol{X}\beta, \Sigma). \tag{17}$$

  for some unspecified covariance matrix Σ.

- Our conditional posterior of β becomes:

$$\beta \mid \Sigma, \boldsymbol{y} \sim \mathrm{N}((\boldsymbol{X}'\Sigma^{-1}\boldsymbol{X})^{-1}\boldsymbol{X}'\Sigma^{-1}\boldsymbol{y}, (\boldsymbol{X}'\Sigma^{-1}\boldsymbol{X})^{-1}) \tag{18}$$

Informative priors

- The results so far all involve flat priors. Now imagine we have informative priors:

$$\boldsymbol{\beta} \sim \mathrm{N}(\boldsymbol{\beta}_0, \boldsymbol{\Sigma}_\beta)$$
$$\sigma^2 \sim \text{Inv-Gamma}(a_0, b_0)$$

  or, for unspecified $\boldsymbol{\Sigma}$,

$$\boldsymbol{\Sigma} \sim \text{Inv-Wishart}(\nu_0, \boldsymbol{S}_0).$$

- We can modify our previous results to include informative priors.

Informative priors

- To include informative priors on β, define a new regression model

$$\boldsymbol{y}^* \sim \mathrm{N}(\boldsymbol{X}^*\beta, \boldsymbol{\Sigma}^*)$$

  where the matrices with $*$ augment data with priors:

$$\begin{pmatrix} \boldsymbol{y} \\ \beta_0 \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} \boldsymbol{X} \\ \boldsymbol{I} \end{pmatrix} \beta, \begin{pmatrix} \boldsymbol{\Sigma} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{\Sigma}_\beta \end{pmatrix} \right).$$

- Now sample from this model as if we had flat priors!

Informative priors

- Turning to σ^2, we now have

$$\sigma^2 \mid \boldsymbol{y}, \beta \sim \text{Scale-inv-}\chi^2(a_0 + n, b_0 + \frac{1}{n}(\boldsymbol{y} - \boldsymbol{X}\beta)'(\boldsymbol{y} - \boldsymbol{X}\beta)). \tag{19}$$

- Or, if we had an unrestricted covariance matrix $\boldsymbol{\Sigma}$,

$$\boldsymbol{\Sigma} \mid \boldsymbol{y}, \beta \sim \text{inv-Wishart}(\nu_0 + n, \boldsymbol{S}_0 + (\boldsymbol{y} - \boldsymbol{X}\beta)(\boldsymbol{y} - \boldsymbol{X}\beta)'). \tag{20}$$

Extensions to SEM

SEM

- Idea of Gibbs sampler for SEM: try to make the SEM look like a regression model, so that we can use our regression results.
  - If we sample our latent variables and condition on them, then we are close to regression.
  - So we will sample parameters in three steps: latent variables, then "location" variables (means, intercepts, loadings, regression weights), then "scale" variables (variances and covariances).

Latent Variables

- We can derive the conditional distribution of $\boldsymbol{\eta}_i$ given $\boldsymbol{y}_i$ and all other model parameters. Idea:
  - The $\boldsymbol{\eta}_i$ and $\boldsymbol{y}_i$ are jointly distributed as multivariate normal. The means and covariances can be found using properties of expected values and covariances.
  - Once we have the joint distribution, we can obtain the conditional distribution using known results of the multivariate normal.

Latent Variables

- Using this strategy, the distribution of $\boldsymbol{\eta}_i$ given $\boldsymbol{y}_i$ and other model parameters is multivariate normal with mean

$$\left(\boldsymbol{\Lambda}'\boldsymbol{\Theta}^{-1}\boldsymbol{\Lambda} + \left((\boldsymbol{I}-\boldsymbol{B})^{-1}\boldsymbol{\Psi}(\boldsymbol{I}-\boldsymbol{B}')^{-1}\right)^{-1}\right)^{-1}\left(\boldsymbol{\Lambda}'\boldsymbol{\Theta}^{-1}(\boldsymbol{y}_i-\boldsymbol{\nu}) + (\boldsymbol{I}-\boldsymbol{B}')\boldsymbol{\Psi}^{-1}\boldsymbol{\alpha}\right) \tag{21}$$

and the covariance matrix is

$$\left(\boldsymbol{\Lambda}'\boldsymbol{\Theta}^{-1}\boldsymbol{\Lambda} + \left((\boldsymbol{I}-\boldsymbol{B})^{-1}\boldsymbol{\Psi}(\boldsymbol{I}-\boldsymbol{B}')^{-1}\right)^{-1}\right)^{-1}. \tag{22}$$

Location Parameters

- To sample the location parameters (call them $\boldsymbol{\xi}$), we rewrite our model so that it is a regression model. This involves treating the $\boldsymbol{\eta}_i$ as data and the location parameters as regression weights.
- The regression model design matrix includes entries of 0, 1, and latent variables. The model covariance matrix includes the residual covariance matrices $\boldsymbol{\Theta}$ and $\boldsymbol{\Psi}$.

Location Parameters

- Example of a CFA with three observed variables and one latent variable:

$$\begin{pmatrix} y_{i1} \\ y_{i2} \\ y_{i3} \\ \eta_{i1} \end{pmatrix} \sim \mathrm{N}\left(\begin{pmatrix} 1 & \eta_{i1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \eta_{i1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \eta_{i1} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \nu_1 \\ \lambda_1 \\ \nu_2 \\ \lambda_2 \\ \nu_3 \\ \lambda_3 \end{pmatrix}, \boldsymbol{V} \right)$$

with

$$\boldsymbol{V} = \begin{pmatrix} \boldsymbol{\Theta} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Psi} \end{pmatrix}. \tag{23}$$

Location Parameters

- The previous slide involves only one person/case i. We need to use all the people for our sampler.
- We stack people on top of each other to get

$$\begin{pmatrix} \boldsymbol{y}_1 \\ \boldsymbol{\eta}_1 \\ \boldsymbol{y}_2 \\ \boldsymbol{\eta}_2 \\ \vdots \\ \boldsymbol{y}_n \\ \boldsymbol{\eta}_n \end{pmatrix} \sim \mathsf{N}\left(\begin{pmatrix} \boldsymbol{H}_1 \\ \boldsymbol{H}_2 \\ \vdots \\ \boldsymbol{H}_n \end{pmatrix} \begin{pmatrix} \nu_1 \\ \lambda_1 \\ \nu_2 \\ \lambda_2 \\ \nu_3 \\ \lambda_3 \end{pmatrix}, \boldsymbol{I} \otimes \boldsymbol{V} \right),$$

or more concisely,

$$\boldsymbol{z} \sim \mathsf{N}(\boldsymbol{H}\boldsymbol{\xi}, \boldsymbol{I} \otimes \boldsymbol{V}), \tag{24}$$

Location Parameters

- Now we can apply our regression Gibbs sampler to the previous slide. Using Equation (18), we have:

$$\boldsymbol{\xi} \mid \boldsymbol{H}, \boldsymbol{V} \sim \mathrm{N}\left((\boldsymbol{H}'(\boldsymbol{I} \otimes \boldsymbol{V})^{-1}\boldsymbol{H})^{-1}\boldsymbol{H}'(\boldsymbol{I} \otimes \boldsymbol{V})^{-1}\boldsymbol{z}, (\boldsymbol{H}'(\boldsymbol{I} \otimes \boldsymbol{V})^{-1}\boldsymbol{H})^{-1}\right). \quad (25)$$

- And if we use normal priors $\boldsymbol{\xi} \sim \mathrm{N}(\boldsymbol{\xi}_0, \boldsymbol{\Sigma}_\xi)$, we can use the same regression trick of appending the priors to the end.

Location Parameters

- Equation (25) involves inverting some very large block diagonal matrices. The block diagonal property allows for simplifications that lead to major computational speedup:

$$\boldsymbol{\xi} \mid \boldsymbol{H}, \boldsymbol{V} \sim \mathrm{N}(\boldsymbol{D}\boldsymbol{d}, \boldsymbol{D}) \tag{26}$$

$$\boldsymbol{D} = \left(\boldsymbol{\Sigma}_{\xi}^{-1} + \sum_{i=1}^{n} \boldsymbol{H}_i' \boldsymbol{V}^{-1} \boldsymbol{H}_i \right)^{-1} \tag{27}$$

$$\boldsymbol{d} = \boldsymbol{\Sigma}_{\xi}^{-1} \boldsymbol{\xi}_0 + \sum_{i=1}^{n} \boldsymbol{H}_i \boldsymbol{V}^{-1} \boldsymbol{z}_i. \tag{28}$$

- These results are related to those described by Asparouhov & Muthén (2010).

Scale Parameters

- The only parameters left are the residual variances and covariances in $\boldsymbol{\Theta}$ and $\boldsymbol{\Psi}$.
- We continue to follow regression here. We get residuals of $\boldsymbol{y}_i$ and of $\boldsymbol{\eta}_i$, and those residuals play a major role in the posterior sampling.
- For the Gibbs sampling results below, our covariance matrices need to be unrestricted or block diagonal.

Priors

- Two situations for priors:
  - A standalone variance, say ψ_{jj}, has an Inverse Gamma(a_j, b_j) prior.
  - A block of the covariance matrix, say $\boldsymbol{\Psi}_k$, has an Inverse Wishart(ν_k, $\boldsymbol{S}_k$) prior.

Posteriors

- This leads to two situations for posteriors.
- For a standalone variance ψ_{jj}, the posterior is

$$\psi_{jj} \mid \boldsymbol{\xi}, \boldsymbol{\eta} \sim \text{Scale-inv-}\chi^2 \left(a_j + n, b_j + \frac{1}{n} \sum_{i=1}^{n} (\eta_{ij} - (\alpha_j + \sum_{k=1}^{m} B_{jk} \eta_{ik}))^2 \right). \quad (29)$$

Posteriors

- For a block $\boldsymbol{\Psi}_k$, define a residual matrix

$$\boldsymbol{E} = \sum_{i=1}^{n} (\boldsymbol{\eta}_i - (\boldsymbol{\alpha} + \boldsymbol{B}\boldsymbol{\eta}_i))(\boldsymbol{\eta}_i - (\boldsymbol{\alpha} + \boldsymbol{B}\boldsymbol{\eta}_i))'.$$

- Then the posterior is

$$\boldsymbol{\Psi}_k \mid \boldsymbol{\xi}, \boldsymbol{\eta} \sim \text{Inv-Wishart}(\nu_k + n, \boldsymbol{S}_k + \boldsymbol{E}_k), \tag{30}$$

where $\boldsymbol{E}_k$ is the block of the matrix $\boldsymbol{E}$ that corresponds to $\boldsymbol{\Psi}_k$.

Posteriors

- What about the residual covariance matrix of observed variables, $\boldsymbol{\Theta}$? The results are nearly identical. We just define the residual $\boldsymbol{E}$ matrix as

$$\boldsymbol{E} = \sum_{i=1}^{n} (\boldsymbol{y}_i - (\boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\eta}_i))(\boldsymbol{y}_i - (\boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\eta}_i))'.$$

Outline of Gibbs Sampler

Outline

- Assign starting values to all model parameters.
- For many thousands of iterations:
  - Sample latent variables using Equations (21) and (22), treating all other parameters as known.
  - Sample location parameters using Equation (26), treating all other parameters as known.
  - Sample scale parameters using Equations (29) and (30), treating all other parameters as known.

Programming Considerations

- We can use the *lavaan* model specification syntax to help automate these steps:
  - First run a *lavaan* model with `do.fit = FALSE`. Call the resulting object `fit`.
  - `lavInspect(fit, 'data')` returns the dataset used in your model.
  - `lavInspect(fit, 'free')` returns model matrices, where nonzero entries correspond to free parameters.
  - `lavInspect(fit, 'est')` returns model matrices, where nonzero entries include fixed values and starting values of free parameters.

Programming Considerations

- Some tricky steps remain:
  - Set up the regression-like model for sampling location parameters.
  - Handle equality constraints.
  - Handle standalone observed variables (latent variable with variance of 0).
  - Handle residual covariances. (May require Metropolis-Hastings to maintain a positive definite covariance matrix.)

Summary

- The Gibbs sampler for traditional SEM is heavy on notation. But the underlying ideas are similar to regression.
- The fact that we condition on sets of parameters (latent variables, location parameters, scale parameters) is helpful for model extension. We could change the model for one set of parameters, and the sampling algorithm for other sets of parameters stays intact.
- Example: I want a nonnormal latent variable distribution. If I can find a way to sample the nonnormal latent variables, then I can sample the model's location and scale parameters in the usual way.

Transition

- This Gibbs sampler is not very efficient. It can take it a long time to adequately explore the posterior distribution.
- In practice, this translates to obscene amounts of samples from the posterior distribution. It is not unusual to run a Gibbs sampler for 50,000 or 100,000 iterations.
- Newer MCMC methods, such as Hamiltonian Monte Carlo, are much more efficient. This means we don't need as many posterior samples. Details are in the next section.

Part 4: Hamiltonian Monte Carlo

Hamiltonian Monte Carlo

- In the past decade, Hamiltonian Monte Carlo has been popularized via Stan and related software.
- Here, we consider how we could develop our own HMC samplers outside of Stan.

Why HMC

- Gibbs samplers are flexible and can handle many of our models. Why HMC?
  - For many models, HMC can more efficiently explore the posterior distribution because it uses the model gradient. This means we can make accurate inferences using fewer posterior samples.
  - Scalability: Potential of handling large models with many parameters and huge sample sizes.
  - Increased caution: HMC often lets you know when you made a mistake.

Outline of Part 4

- The goal of this part is to learn about HMC and get an idea about how to code our own HMC samplers.
  - HMC background and intuition
  - Implementation of HMC for psychometric models
  - Extensions to other psychometric models

Background and Intuition

Intuition

- Recall our simple, two parameter posterior distribution that looked like a mountain.
- Now turn the mountain upside down so it is a bowl.
- Now roll a marble around the bowl. The path of the marble is related to the posterior samples that we will draw.

Momentum

- How can we conceptualize movement around the posterior distribution? We need to define a momentum corresponding to each dimension (parameter) of our posterior distribution.
- Momentum examples involving two-dimensional posterior distribution:
  - $(10, 0)$ means that we are moving from left to right along the x-axis (the Intercept axis), but we are not moving at all along the y-axis (the Slope axis)
  - $(-10, 5)$ means that we are moving from right to left along the x-axis, and we are moving from bottom to top of the y-axis at a slower rate.
  - Similar to an Etch-a-Sketch toy!

Hamiltonian

- HMC involves ideas from physics. First, the *Hamiltonian* describes the total amount of energy in our marble. This consists of:
  - Kinetic energy, which involves the momentum along each dimension (collected in a vector $\boldsymbol{m}$).
  - Potential energy along each dimension, which is related to the height of the bowl at the marble's current location.
- The total amount of energy (kinetic plus potential) is conserved in the system.

Hamiltonian

- Assume that the marble's starting momentum is drawn from a multivariate normal:

$$\boldsymbol{m} \sim \mathrm{N}(\mathbf{0}, \boldsymbol{M}),$$

where $\boldsymbol{M}$ is a tuning parameter.

- Then our Hamiltonian is

$$H(\boldsymbol{\theta}, \boldsymbol{m}) = -\log p(\boldsymbol{\theta} \mid \boldsymbol{Y}) + \frac{1}{2}\boldsymbol{m}'\boldsymbol{M}^{-1}\boldsymbol{m}, \tag{31}$$

where $\boldsymbol{\theta}$ is the model parameter vector, $\boldsymbol{Y}$ is the observed data, $p(\boldsymbol{\theta} \mid \boldsymbol{Y})$ is the model posterior distribution.

Hamilton's equations

- The Hamiltonian describes the energy at a particular parameter value $\boldsymbol{\theta}$ with a particular momentum $\boldsymbol{m}$. To describe movement through the posterior distribution, we need to describe how $\boldsymbol{\theta}$ and $\boldsymbol{m}$ jointly change over time t. This leads to Hamilton's equations:

$$\frac{\partial \boldsymbol{m}}{\partial t} = -\frac{\partial H(\boldsymbol{\theta}, \boldsymbol{m})}{\partial \boldsymbol{\theta}} = \frac{\partial \log p(\boldsymbol{\theta} \mid \boldsymbol{Y})}{\partial \boldsymbol{\theta}} \tag{32}$$

$$\frac{\partial \boldsymbol{\theta}}{\partial t} = \frac{\partial H(\boldsymbol{\theta}, \boldsymbol{m})}{\partial \boldsymbol{m}} = \boldsymbol{M}^{-1}\boldsymbol{m}, \tag{33}$$

- A solution to these equations is a path by which the marble travels around the posterior. (Values of $\boldsymbol{\theta}$ and $\boldsymbol{m}$ jointly changing over time.)

HMC

- We can now start to see how the HMC sampler works:
  - Define model log-likelihood and priors, draw a momentum vector $\boldsymbol{m}$.
  - Solve Hamilton's equations, defining a path around the posterior distribution.
  - Stop after some amount of time, with the value of $\boldsymbol{\theta}$ at the stopping point being a posterior sample.

Leapfrog

- The "solve Hamilton's equations" part is not so simple. The equations cannot be solved analytically.
- Instead, we use a *leapfrog integrator* to approximate a solution.
- Write $\boldsymbol{m}(t)$ and $\boldsymbol{\theta}(t)$ as the momentum and parameter values at time t. That is, as t changes, $\boldsymbol{m}$ and $\boldsymbol{\theta}$ also change.

Leapfrog

- One leapfrog step of size ϵ is

$$\begin{aligned}
\boldsymbol{m}(t+\epsilon/2) &= \boldsymbol{m}(t) + (\epsilon/2)\frac{\partial \log p(\boldsymbol{\theta}(t) \mid \boldsymbol{Y})}{\partial \boldsymbol{\theta}(t)} \\
\boldsymbol{\theta}(t+\epsilon) &= \boldsymbol{\theta}(t) + \epsilon \boldsymbol{M}^{-1}\boldsymbol{m}(t+\epsilon/2) \\
\boldsymbol{m}(t+\epsilon) &= \boldsymbol{m}(t+\epsilon/2) + (\epsilon/2)\frac{\partial \log p(\boldsymbol{\theta}(t+\epsilon) \mid \boldsymbol{Y})}{\partial \boldsymbol{\theta}(t)},
\end{aligned}$$

  where ϵ is the step size.

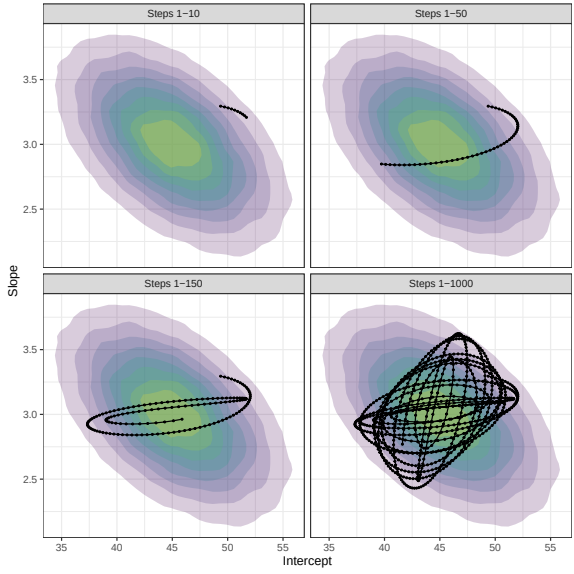
Leapfrog integrator

```
leapfrog <- function(theta, momentum, grfun, data, eps, Minv) {
  newmom <- momentum + (eps/2) * grfun(theta, data)
  newtheta <- theta + eps * Minv %*% newmom
  newmom <- newmom + (eps/2) * grfun(newtheta, data)

  list(theta = newtheta, momentum = newmom)
}
```

Sampler

- Refining the steps of our HMC sampler:
  - Define model log-likelihood and priors, set tuning parameters $\boldsymbol{M}$ and ϵ, set number of leapfrog steps per iteration n_ϵ, set initial values for $\boldsymbol{\theta}$.
  - For each iteration:
    - Draw $\boldsymbol{m}(0) \sim \mathrm{N}(\mathbf{0}, \boldsymbol{M})$.
    - Take n_ϵ leapfrog steps, updating $\boldsymbol{m}$ and $\boldsymbol{\theta}$ at each step. This leads to $\boldsymbol{m}(n_\epsilon \times \epsilon)$ and $\boldsymbol{\theta}(n_\epsilon \times \epsilon)$.
    - Accept $\boldsymbol{\theta}(n_\epsilon \times \epsilon)$ with some probability (related to Metropolis-Hastings).
    - Repeat, using the current value of $\boldsymbol{\theta}$ as the starting value of the next iteration.

Example


Steps 1–10
Steps 1–50
Steps 1–150
Steps 1–1000
Slope
Intercept
3.5
3.0
2.5
35
40
45
50
55

Example

- The figure shows that it is important to determine a good stopping point: at 10 steps, we haven't gone very far. After hundreds of steps, we are revisiting where we have already been.

- The "No U-Turn" method of Hoffman & Gelman (2014) automatically determines a stopping point. We take additional steps until the trajectory turns back on itself, which happens when

$$(\boldsymbol{\theta}(t) - \boldsymbol{\theta}(0))' \boldsymbol{m}(t) < 0, \tag{34}$$

Example

- The figure also helps us explain some issues surrounding HMC:
  - We cannot have discrete parameters because we cannot travel over cliffs.
  - Parameters should have no bounds, otherwise we may hit a wall. (Stan automatically "unbounds" parameters before sampling.)
  - The step size ϵ needs to be reasonable. (Stan adaptively sets the step size.)

Example

- The figure also helps us explain some diagnostic warnings that many of us have seen before:
  - Divergent transition: After taking a number of steps, the value of the Hamiltonian $H()$ has changed significantly. This happens because the leapfrog integrator is a numerical approximation, and it is problematic because $H()$ is supposed to be conserved. It occurs when the geometry of the posterior distribution has varying amounts of curvature, and/or when the step size ϵ is excessively large. It often indicates a problem with the model specification.
  - Maximum treedepth: The leapfrog sampler took the maximum number of steps (which is set in advance) without encountering a U-turn. This can happen when the step size is excessively small.

Implementation

Implementation

- Let's now step back and consider the ingredients of an HMC sampler, which will help us think about implementation:
  - From user: data $\boldsymbol{Y}$, desired model $p(\boldsymbol{Y} \mid \boldsymbol{\theta})$, prior distribution $p(\boldsymbol{\theta})$.
  - From software: functions to evaluate the model log-likelihood and gradient at new values of $\boldsymbol{\theta}$. (Stan uses automatic derivatives here; see Cudeck (2005) for application to psychometrics!)
  - Tuning parameters: Momentum "mass matrix" $\boldsymbol{M}$, step size ϵ, maximum number of leapfrog steps n_ϵ. (can all be automatically or adaptively set in Stan)

Implementation

- Without Stan, how could we obtain these ingredients for psychometric models?
  - Have user specify the model in *(b)lavaan* syntax.
  - Use internal *lavaan* functions to evaluate model log-likelihood and gradient: `lav_model_set_parameters()`, `lav_model_implied()`, `lav_model_gradient()`.
  - Unbound the parameters, which will require an additional chain rule. Evaluate the prior log-densities.
  - Set tuning parameters by hand: $\boldsymbol{M}^{-1}$ should approximate the posterior covariance matrix and can be diagonal for simple computation; ϵ may be set to achieve an optimal acceptance rate; n_ϵ can be handled by the NUTS equation.

Psychometric application

- Focusing on psychometric models, the conditional/marginal likelihood distinction becomes important for HMC. Recall:
- Conditional on the $\boldsymbol{\eta}_i$, we have

$$\boldsymbol{y}_i \mid \boldsymbol{\eta}_i \sim \mathsf{N}(\boldsymbol{\nu} + \boldsymbol{\Lambda}\boldsymbol{\eta}_i, \boldsymbol{\Theta}). \tag{35}$$

- Marginalizing out the $\boldsymbol{\eta}_i$, we have

$$\boldsymbol{y}_i \sim \mathsf{N}(\boldsymbol{\nu} + \boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\alpha}, \boldsymbol{\Lambda}(\boldsymbol{I} - \boldsymbol{B})^{-1}\boldsymbol{\Psi}(\boldsymbol{I} - \boldsymbol{B}')^{-1}\boldsymbol{\Lambda}' + \boldsymbol{\Theta}) \tag{36}$$

Psychometric application

- These are two likelihoods for the same model! We will obtain the same posterior distributions from either!
- Use of the conditional likelihood gives us conditional independence, which often means that we can work with univariate normal likelihoods instead of multivariate normal likelihoods. But we also must sample all the $\boldsymbol{\eta}_i$, which involves sampling hundreds or thousands of extra parameters. Many more gradient computations are required.
- Use of the marginal likelihood allows us to avoid sampling the $\boldsymbol{\eta}_i$, which greatly reduces the number of parameters to sample. But we also need to evaluate multivariate normal likelihoods that may be of high dimension.

Psychometric application

- Merkle, Fitzsimmons, Uanhoro, & Goodrich (2021) compared conditional and marginal approaches to traditional SEMs in Stan. They showed that, for models with a relatively small number of observed variables (up to 11), sampling speed and efficiency are considerably improved when we use the marginal likelihood.
- Information criteria like WAIC (Watanabe, 2010) and PSIS-LOO (Vehtari, Gelman, & Gabry, 2017) are also more stable and theoretically appropriate under the marginal approach (Merkle, Furr, & Rabe-Hesketh, 2019).
- However, it is more difficult to extend the marginal approach to more complex models. When the marginal approach works it is nice, but it may not work for your non-standard model!

Extension to other psychometric models

Extensions

- While we have gotten far in describing the samplers, we have also ignored some situations that often occur in psychometrics:
  - Models of ordinal variables.
  - Models that include discrete parameters.
  - Multilevel SEM.

Ordinal variables

- To handle ordinal variables in an SEM framework, it is useful to consider the data augmentation method of Chib & Greenberg (1998).
- Let y be an ordinal variable with K categories, then assume an underlying continuous variable y^* such that:

$$
\begin{aligned}
y = 1 \text{ if } y^* &< \tau_1 \\
y = 2 \text{ if } \tau_1 < \ & y^* < \tau_2 \\
y = 3 \text{ if } \tau_2 < \ & y^* < \tau_3 \\
&\vdots \\
y = K \text{ if } y^* > \ & \tau_{K-1},
\end{aligned}
$$

where $\tau_1 < \tau_2 < \tau_3 < \ldots < \tau_{K-1}$.

Ordinal variables

- Two options for MCMC of models with ordinal variables:
  - Sample all the y_{ij}^* as extra parameters. Then apply the usual SEM to the y_{ij}^*. (current approach in *blavaan*).
  - Sample the latent variables $\boldsymbol{\eta}_i$. Then use conditional independence to compute the probability of observing each ordered category (which is easy to do with the univariate normal, but difficult with the multivariate normal!).
  - Technical information can be found in this case study.

Ordinal variables

- For models with ordinal variables, we need to sample some extra parameters in $\boldsymbol{y}_i^*$ or $\boldsymbol{\eta}_i$.
  - For models without many observed variables (say, less than 15), Stan works reasonably well by sampling the $\boldsymbol{y}_i^*$.
  - For models with many observed variables, sampling the $\boldsymbol{\eta}_i$ becomes better.
  - In general, we will see less efficiency and longer runtimes as compared to models of continuous variables.

Discrete parameters

- We often consider models with discrete parameters, such as mixture models or cognitive diagnosis models.
- For many models, we can marginalize out the discrete parameters and use the usual HMC methods.

Discrete parameters

- This is easy to see in a two-component mixture of normal distributions. A typical Bayesian specification could involve:

$$
\begin{aligned}
y_i | d_i &\sim \mathrm{N}(\mu_{d_i}, \sigma_{d_i}) \\
d_i &\sim \text{Bernoulli}(p_i) \\
p_i &= \text{logit}^{-1}(x_i \beta)
\end{aligned}
$$

- The d_i are binary so cannot be sampled via HMC.

Marginalization

- We can instead write

$$y_i \sim p_i \mathrm{N}(\mu_1, \sigma_1) + (1 - p_i)\mathrm{N}(\mu_0, \sigma_0)$$
$$p_i = \mathrm{logit}^{-1}(x_i \beta)$$

- We no longer need the discrete parameter d_i, so we can use HMC.

Marginalization

- This marginalization strategy may be difficult or impossible for models with many discrete parameters. Cognitive diagnosis models are a good example.
- Here, it may be possible to use HMC within Gibbs sampling: use HMC to sample continuous parameters conditioned on discrete parameters, then Gibbs or Metropolis Hastings to sample discrete parameters conditioned on continuous parameters.
- See Martinez & Templin (2023) for progress on this idea.

Multilevel SEM

- Multilevel SEM typically involves latent variables at two or more levels.
- Example: Each student provides multiple test scores, and students are nested in schools. We have latent variable(s) for each student, and latent variable(s) for each school.
- The conditional/marginal distinction again becomes important for HMC efficiency.

Multilevel SEM

- Conditional approach (most popular): Sample the student latent variables and school latent variables. Conditional on these latent variables, the data are (usually) independent. This means our model log-likelihood is a sum of univariate normal distributions.
- Marginal approach: Do not sample latent variables. Use frequentist results from McDonald & Goldstein (1989) and others to efficiently evaluate high-dimensional multivariate normal distributions. (see Rosseel, 2021)

Multilevel

- Example: Say that we have a random-intercept CFA with 20 students in each of 250 schools (so 5,000 students total). The model has one student latent variable; one school latent variable; 4 observed variables contributed by each student.
  - Traditional MCMC: Sample 5,250 latent variables, so that we can evaluate 20k univariate normals.
  - blavaan/Stan: Evaluate 250 multivariate normals, each of dimension 80. Evaluate the likelihood of the 80-dimensional normal by computing inverses/determinants of 4×4 matrices.

Summary

- HMC has many moving parts, but we the basic pieces are not too computationally difficult. There is room to consider custom HMC samplers for psychometric models.
- For psychometric models, it is beneficial to consider using the marginal likelihood. This is somewhat unusual in Bayesian modeling: Bayesians typically sample latent variables, and frequentists avoid latent variables.
- *lavaan* already supplies a good deal of functionality that is required of an HMC sampler.

Conclusions and Final Thoughts

Workflow

- Some general workflow recommendations:
  - Use frequentist models as quick checks of syntax/model ideas.
  - Consider your prior distributions, and do prior predictive checks.
  - Arguments `burnin = 100, sample = 100` are sufficient for rough results of many *blavaan* models.
  - Consider meaningful summaries of how the posterior distribution corresponds to the observed data, based on the goals of your model.

Workflow

- Related to the previous slide: tailor models and summaries to your substantive goals.
  - Use traditional SEM as a launching point for tailored models and tailored model checks.
  - Posterior samples allow us to summarize uncertainty in most quantities of interest (e.g., item-total correlations).
  - Recipes can be helpful, but also limiting. Some coding skills can take you far.

Limitations

- The examples presented here "just worked." We continue to improve *blavaan* to "just work" on more datasets. Some models/datasets currently won't work so well, including:
  - Observed variables that assume large values, with no consideration of priors (lack of model convergence).
  - Large amounts of missing data, large numbers of observed variables (slow).
  - Inclusion of reverse-coded items: must carefully consider whether priors on loadings match sign constraints on loadings.

Questions

- Questions/comments/discussion

Thank you!

In R:

```
install.packages("blavaan")
```

blavaan website:

https://ecmerkle.github.io/blavaan/

References I

Asparouhov, T., & Muthén, B. (2010). Bayesian analysis using Mplus: Technical implementation.

Bonifay, W., & Depaoli, S. (2021). Model evaluation in the presence of categorical data: Bayesian model checking as an alternative to traditional methods. *Prevention Science*, *24*(3), 467–479. http://doi.org/10.1007/s11121-021-01293-w

Chib, S., & Greenberg, E. (1998). Analysis of multivariate probit models. *Biometrika*, *85*, 347–361.

Cudeck, R. (2005). Fitting psychometric models with methods based on automatic differentiation. *Psychometrika*, *70*(4), 599–617. http://doi.org/10.1007/s11336-005-1404-9

De Boeck, P., & Wilson, M. (2004). *Explanatory item response models: A generalized linear and nonlinear approach*. New York: Springer-Verlag.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., & Rubin, D. B. (2013). *Bayesian data analysis* (3rd ed.). Chapman & Hall/CRC.

References II

Hoffman, M. D., & Gelman, A. (2014). The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, *15*(47), 1593–1623. Retrieved from http://jmlr.org/papers/v15/hoffman14a.html

Martinez, A. J., & Templin, J. (2023). Estimating Bayesian diagnostic models with attribute hierarchies with the Hamiltonian-Gibbs hybrid sampler. *Multivariate Behavioral Research*, *58*(1), 141–142. http://doi.org/10.1080/00273171.2022.2160950

McDonald, R. P. (1999). *Test theory: A unified treatment*. Mahwah, NJ: Erlbaum.

McDonald, R. P., & Goldstein, H. (1989). Balanced versus unbalanced designs for linear structural relations in two level data. *British Journal of Mathematical and Statistical Psychology*, *42*, 215–232.

Merkle, E. C., Ariyo, O., Winter, S. D., & Garnier-Villarreal, M. (2023). Opaque prior distributions in Bayesian latent variable models. *Methodology*, *19*(3), 228–255. http://doi.org/10.5964/meth.11167

References III

Merkle, E. C., Fitzsimmons, E., Uanhoro, J., & Goodrich, B. (2021). Efficient Bayesian structural equation modeling in Stan. *Journal of Statistical Software*, *100*(6), 1–22. Retrieved from https://doi.org/10.18637/jss.v100.i06

Merkle, E. C., Furr, D., & Rabe-Hesketh, S. (2019). Bayesian comparison of latent variable models: Conditional versus marginal likelihoods. *Psychometrika*, *84*, 802–829.

Merkle, E. C., & Rosseel, Y. (2018). blavaan: Bayesian structural equation models via parameter expansion. *Journal of Statistical Software*, *85*(4), 1–30.

Rosseel, Y. (2021). Evaluating the observed log-likelihood function in two-level structural equation modeling with missing data: From formulas to R code. *Psych*, *3*(2), 197–232. http://doi.org/10.3390/psych3020017

Vehtari, A., Gelman, A., & Gabry, J. (2017). Practical Bayesian model evaluation using leave-one-out cross-validation and WAIC. *Statistics and Computing*, *27*, 1413–1432.

Watanabe, S. (2010). Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of Machine Learning Research*, *11*, 3571–3594.