Introduction, Bayes, and Measurement Models

SMIP Summer School 2023: Lecture 1

## Workshop Introduction

This is an ambitious workshop:

* Measurement models (Item Response Theory)
* Multilevel models for clustered data
* Bayesian statistics
* Multilevel measurement models

Each could be its own workshop

## Workshop Materials

Set of slides on topics pulled from other course materials

* In case additional detail on any topic is needed

Full set of analysis files in R and Stan

* In case you would like to work ahead

## Workshop Schedule

* Mornings: 9:00-12:30 (13:00 Friday)
  + Break at 10:45
* Afternoons:
  + Tuesday: 14:00-17:15 (break at 16:00)
  + Wednesday: 15:15-17:45 (break at 16:45)
  + Thursday: 14:00-15:45 (no break)

## Lecture and Example Time Plan

Last hour of each day is open

Your choice:

* Questions
* Example practice
* Work with your data
* Leave early

## Syntax and Model Note

* As stan models can take a very long time to run, all prior model results have been saved
* Model results are in the models folder
* The scriptGuide.md file contains the details for each model
* Some slides are written in Quarto (markdown), which can be rendered from the slides folder

## Workshop Sections

1. Introduction to Measurement Models, Bayesian Statistics, and Stan
2. Introduction to Multilevel Models
3. Introduction to Multilevel Measurement Models

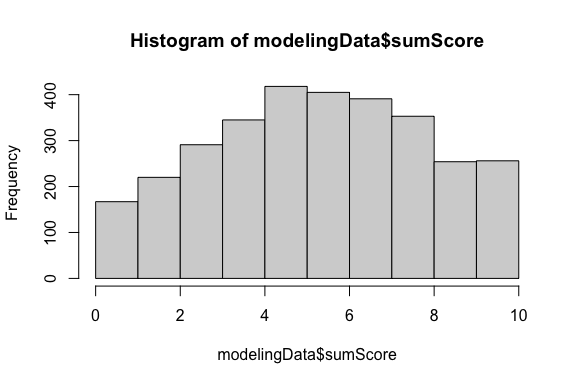
# Running Example

## Running Example Data

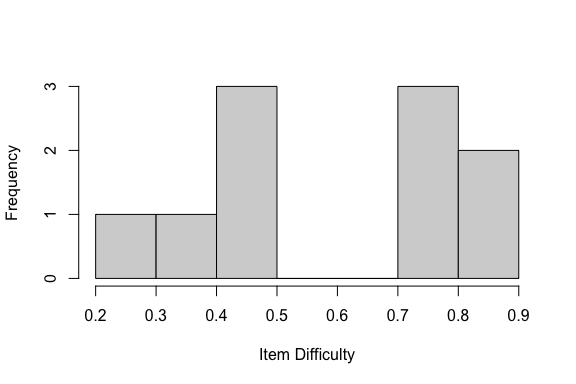
Data come from a secondary school (10th grade) end-of-grade mathematics assessment given around the year 2006 in a midwestern state in the USA

* Sample of 50 students from 62 schools
* Sample of 10 mathematics items from the assessment
  + Items are scored correct/incorrect (score1-score10)
  + The sum of all 10 items is included (sumScore)
* Other variables:
  + studentID: Student ID number (created for this example)
  + schoolID: School ID number (created for this example)
  + frlunch: Free/reduced lunch status (1 = free/reduced lunch, 0 = not free/reduced lunch; indirect indicator of student socioeconomic status)

## Distribution of Sum Scores



## Distribution of Item Difficulties



# Introduction to Measurement Models

## Section Objectives

1. Latent traits
2. Our first graphical model (path diagram)
3. Psychometric models from generalized linear models

## Latent Traits: A Big-Picture View

Latent trait theory posits there are attributes of a person (typically) that are:

* Unobservable (hence the term latent)
* Quantifiable
* Related to tasks that can be observed

Often, these attributes are often called constructs, underscoring they are constructed and do not exist, such as:

* A general or specific ability (in educational contexts)
* A feature of personality, such as “extroversion” (in psychological contexts)

The same psychometric models apply regardless of the measurement context

## Latent Traits are Interdisciplinary

* Many varying fields use some version of latent traits
* Similar (or identical) methods are often developed separately
  + Item response theory in education
  + Item factor analysis in psychology
* Many different terms for same ideas, such as the
  + Label given to the latent trait: Factor/Ability/Attribute/Construct
  + Label given to those giving the data: Examinee/Subject/Participant/Respondent/Patient/Student
* What this means:
  + Lots of words to keep track of, but (relatively) few concepts
  + We will focus on concepts (but have a lot of words)

## Best Measures are Built Purposefully

* Latent constructs seldom occur randomly—they are defined
  + The definition typically indicates:
    - What the construct means
    - What observable behaviors are likely related to the construct
      * For a lot of what we do, observable behavior means answering questions on an assessment or survey
* Therefore, modern psychometric methods are built around specifying the set of observed variables to which a latent variable relates
  + No need for exploratory analyses—we define our construct and seek to falsify our definition
* The term I use for “relates” is “measure”
  + e.g., Educational assessment items measure some type of ability

## Guiding Principles

* To better understand psychometric methods and theory, envision what analyses would be used if latent variables were not latent
  + Example: Imagine if we could directly observe mathematics ability
* Then, consider what we would do with that value
  + Example: We could predict how students would perform on items using logistic regression (with as an observed predictor)
* Psychometric models essentially do this—use observed variable methods as if we know the value of the latent variable
  + But, with:
    - A data collection design allowing for such methods to be used
    - A more formal vetting of whether or not we did a good job measuring the construct

## Measurement of Latent Constructs

How does this process differ when we cannot observe the thing we are measuring—when the construct is latent?

* We still need something we can observe—item responses for example
* We need a method to map the response to a number (Strongly agree==5?)
* We also need a way to aggregate all responses to a value that represents a person
  + A score or classification
* We then need a way to ensure what we just did means what we think it does
  + Methods for validation
* We also need to remember that the values we estimate for a person’s latent trait(s) won’t be perfectly reliable
  + Caution needed for secondary analyses

## Measurement Models

* A distinguishing feature of psychometric models is the second word—they are models
  + We often call such models “measurement models”
* Measurement models are the mathematical specification that provides the link between the latent variable(s) and the observed data
* The form of such models looks different across the wide classes of measurement models (e.g., factor analysis vs. item response models) but wide generalities exist
* Measurement models need:
  + Distributional assumptions about the data (with link functions)
  + A linear or non-linear form that predicts data from the trait(s)
* The key: Observed data are being predicted by latent variable(s)

## Measurement Models vs. Other Measurement Techniques

Measurement models are a different way of thinking about psychometrics than what most people without psychometric training do

* Most scientists enumerate item response scores (e.g., correct response == 1; strongly agree == 5)
* The latent trait score estimate is then formed by adding the response scores together [[1]](#footnote-43)
* As it turns out, the naïve adding together of item scores implies a measurement model
  + Called parallel items — very strict assumptions (equal variances and covariances for all observed variables)

## Characteristics of Latent Variables

Latent variables can be defined to have different levels of measurement

* Interval level (as in factor analysis and item response theory) — Continuous
  + No absolute zero; units of the factor are equivalent across the range of values
  + Example: A person with a value of 2 is the same distance from a person with a value of 0 as is a person with a value of -2
* Ordinal level (as in diagnostic classification models)
  + Can rank order people but not determine how far apart they may be
  + Example: Students considered masters of a topic have more ability than students considered non-masters
* Nominal level (as in latent class or finite mixture models) — Categorical
  + Groups/classes/clusters of people

## Most Common: Continuous Latents

* For most of this workshop, we will treat latent variables as continuous (interval level)
* As they do not exist, continuous latent variables need a defined metric:
  + What is their mean?
  + What is their standard deviation?
* Defining the metric is the first step in a latent variable model
  + Called scale identification
* The metric is arbitrary
  + Can set differing means/variances but still have same model
  + Linear transformations of parameters based on scale mean and standard deviation

## Measurement Model Path Diagrams

Measurement models are often depicted in graphical format, using what is called a path diagram

* Typically, latent variables are represented as objects that are circles/ovals
* Using graph theory terms, a variable in a path diagram (latent or observed) is called a node
* Lines connecting the variables are called edges

## Latent Variable Only

if (!require(pathdiagram)) install.packages("pathdiagram")

Loading required package: pathdiagram

Loading required package: shape

library(pathdiagram)  
wall()  
  
latentVariable = latent("Latent Variable", rx= .2, ry=.2)  
draw(latentVariable)



## Adding Observed Variables

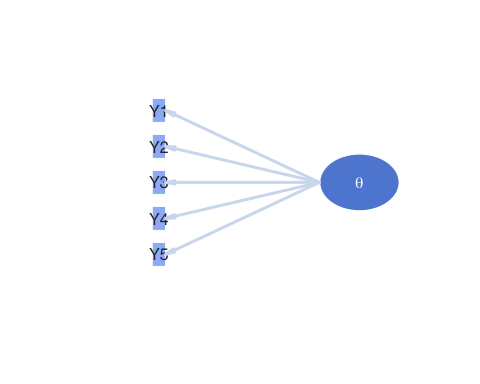
Measurement model path diagrams often denote observed variables with rectangular boxes

On the next slide:

* The term “latent variable” is replaced with
* The observed variables are denoted as through
  + Imagine these represent five observed items of a scale measuring

## Path Diagram with Observed and Latent Variables

if (!require(pathdiagram)) install.packages("pathdiagram")  
wall(ylim=c(.2,1))  
  
latentVariable = latent(expression(theta), rx= .12, ry=.12, x=.8, y = .6)  
draw(latentVariable)  
obsVariables = list()  
for (var in 1:5){  
 obsVariables[[paste0("Y", var)]] = manifest(paste0("Y", var), x = .2 , y=.9-(var-1)\*.15)   
 draw(obsVariables[[paste0("Y", var)]])  
 arrow(from=latentVariable, to = obsVariables[[paste0("Y", var)]], start = "west", end="east")  
}



## Path Diagrams: Not Models

Path diagrams are useful for depicting a measurement model but are not isomorphic with the mathematical models they depict

* All model parameters are often not included in the diagram
* No indication about the distribution of the variables

## Translating a Path Diagram to a Model

Going back to the point from before—let’s imagine the latent variable as an observed variable

* An arrow (edge) indicates one variable predicts another
  + The predictor is the variable on the side of the arrow without the point
  + The outcome is the variable on the side of the point
* If we assume the items were continuous (like linear regression), the diagram indicates a regression model for each outcome

## Interpreting the Parameters

All five regression lines implied by the model are then:

Here:

* is the intercept of the regression line predicting the score from item
  + The expected resposne score for a person who has
* is the slope of the regression line predicting the score from item
  + The expected change in the response score for a one-unit change in

## More Interpreting the Parameters

Also:

* is the residual (error), indicating the difference in the predicted score for person to item
  + Like in regression, we additionally assume:
    - : is normally distributed with mean zero
* is the residual variance of item , indicating the square of how far off the prediction is on average

The five regression models are estimated simultaneously:

* If were observed, we would call this a multivariate regression
  + Multivariate regression: Multiple continuous outcomes predicted by one or more predictors

## More About Regression

In the regression model for a single variable, what distribution do we assume about the outcome?

* As error is normally distributed, the outcome takes a normal distribution
* As , , and are constants, they move the mean of the outcome to
* As error has a variance of , the outcome is assumed to have variance
* Therefore, we say follows a conditionally normal distribution

## The Univariate Normal Distribution

that implies a probability density function (pdf)

* Here, is the constant 3.14 and is Euler’s constant (2.71)
* Of note here is that there are three components that go into the function:
  + The data
  + The mean — this can be the conditional mean we had on the previous slide (formed by parameters)
  + The variance
* The key to using Bayesian methods is to know the distributions for each of the variables in the model

## From Regression to CFA

When is latent, the five-variable model becomes a confirmatory factor analysis (CFA) model

* CFA: Prediction of continuous items using linear regression with one or more continuous latent variables as predictors
  + The interpretations of the regression parameters are identical between linear regression and CFA

## Regression and CFA Differences

The differences between CFA and regression are:

* as a predictor is not observed in CFA but is observed in regression
  + Therefore, we must set its mean and variance
    - There are multiple was to do this (standardized factor, marker item, etc…)—stay tuned
* Each of the model parameters has a different name (and symbol denoting it) in CFA
  + is the item intercept
  + is the factor loading for an item
  + is the unique variance for an item
* We must have a sufficient number of observed variables to empirically identify the latent trait

## Changing Notation

Our five-item CFA model with CFA-notation:

## Measurement Models for Different Item Types

* The CFA model assumes (1) continuous latent variables and (2) continuous item scores
  + What should we do if we have binary items (e.g., yes/no, correct/incorrect)?
* If we had observed and wanted to predict what type of analysis would we use?
* Logistic regression:

## Interpreting Model Parameters

Here:

* is the intercept — the expected log odds of a correct response when
* is the slope — the expected change in log odds of a correct response for a one-unit change in
* Note: there is no error variance term

## Bernoulli Distributions

* Using logistic regression for binary outcomes makes the assumption that the outcome follows a (conditional) Bernoulli distribution, or
  + The parameter is the probability that Y equals one, or
* The Bernoulli pdf (sometimes called the probability mass function as the variable is discrete) is:
* So, there is no error variance parameter in logistic regression as there is no parameter in the distribution that represents error (it is a non-constant function of the mean)
  + Error is represented by how far off a probability is from either zero or one

## Logistic Regression with Latent Variable(s)

* Back to our running example, if we had binary items and wished to form a (unidimensional) latent variable model, we would have something that looked like:

## Logistic Regression with Latent Variable(s)

* Here, the parameters retain their names from CFA:
  + is the item intercept
  + is the factor loading for an item
* We call this slope-intercept parameterization
* This parameterization is called item factor analysis(IFA)
  + Sometimes the intercept is replaced with a threshold (where )

## From IFA to IRT

* IFA and IRT are equivalent models—their parameters are transformations of each other:

## More Comparisons

|  |
| --- |
| Comparing IFA and IRT |

## From IFA to IRT

* This yields the discrimination difficulty parameterization that is common in unidimensional IRT models:
* Here:,
  + is the item difficulty—the point on the scale at which a person has a 50% chance of answering with a one
  + is the item discrimination—the slope of a line tangent to the curve at the item difficulty
* IRT models have a number of different forms of this equation (this is the two-parameter logistic 2PL model)

## IRT Example: Acheivement Data

To demonstrate a couple IRT models, we will compare a 1PL and 2PL model for the example data

## Generalized Linear (Psychometric) Models: Summary

* A key to understanding the varying types of psychometric models is that they must map the theory (the right-hand side of the equation—) to the type of observed data (left-hand side of the equation)
* So far we’ve seen two types of data: continuous (with a normal distribution) and binary (with a Bernoulli distribution)
* For each, the right-hand side of the item model was the same
* For the normal distribution:
  + We had an error term but did not transform the right-hand side
* For the Bernoulli distribution:
  + No error term and a function used to transform the right-hand side so that the conditional mean will range between zero and one

## Introduction to Bayesian Statistics

## The Basics of Bayesian Analyses

* Bayesian statistical analysis refers to the use of models where some or all of the parameters are treated as random components
  + Each parameter comes from some type of distribution
* The likelihood function of the data is then augmented with an additional term that represents the likelihood of the prior distribution for each parameter
  + Think of this as saying each parameter has a certain likelihood – the height of the prior distribution
* The final estimates are then considered summaries of the posterior distribution of the parameter, conditional on the data
  + In practice, we use these estimates to make inferences, just as is done when using non-Bayesian approaches (e.g., maximum likelihood/least squares)

## Why are Bayesian Methods Used?

* Bayesian methods get used because of the *relative* accessibility of one method of estimation (MCMC – to be discussed shortly)
* There are four main reasons why people use MCMC:

1. Missing data
2. Lack of software capable of handling large sized analyses (e.g., computational speed)
3. New models/generalizations of models not available in software
4. Philosophical reasons (e.g., Bayesian ideals)

## Perceptions and Issues with Bayesian Methods

* The use of Bayesian statistics has been controversial
  + The use of certain prior distributions can produce results that are biased or reflect subjective judgment rather than objective science
* Most MCMC estimation methods are computationally intensive
  + Many “easy” Bayesian programs take a lot of syntax
* Understanding of what Bayesian methods had been very limited outside the field of mathematical statistics
* Over the past 20 years, Bayesian methods have become widespread – making new models estimable and becoming standard in some social science fields

## How Bayesian Statistics Work

Bayesian methods rely on Bayes’ Theorem

Here:

* is the prior distribution (pdf) of A (i.e., a Bayesian method)
* is the marginal distribution (pdf) of B
* is the conditional distribution (pdf) of B, given A
* is the posterior distribution (pdf) of A, given B

## A Live Bayesian Example

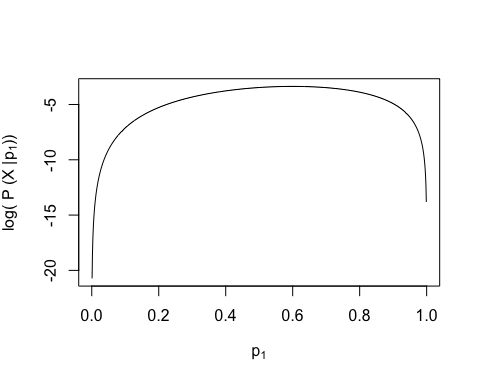
* Suppose we wanted to assess the probability of rolling a 1 on a 6-sided die:
* We then collect a sample of data
  + These are independent tosses of the die
* The posterior distribution of the probability of a 1 conditional on the data is:
* We can determine this via Bayes theorem:

## Defining the Likelihood Function

The likelihood of the data given the parameter:

* Any given roll of the dice is a Bernoulli variable
  + A “success” is defined by rolling a one
* The product in the likelihood function comes from each roll being independent
  + The outcome of a roll does not depend on previous or future rolls

## Visualizing the Likelihood Function



## Choosing the Prior Distribution for

We must now pick the prior distribution of :

* Our choice is subjective: Many distributions to choose from
* What we know is that for a “fair” die, the probability of rolling a one is
  + But…probability is not a distribution
* Instead, let’s consider a Beta distribution

## The Beta Distribution

For parameters that range between zero and one (or two finite end points), the Beta distribution makes a good choice for a prior:

where:

and,

## More Beta Distribution

The Beta distribution has a mean of

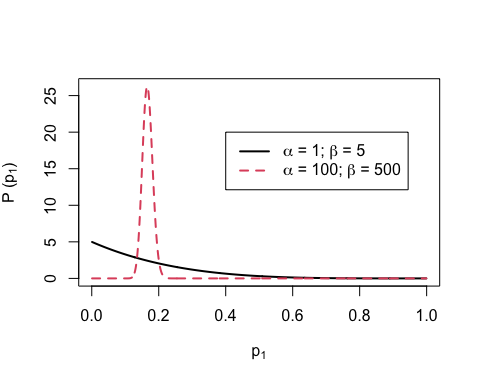
* The parameters and are called hyperparameters
  + Hyperparameters are parameters of prior distributions
* We can pick values of and to correspond to
  + Many choices: and have the same mean as and
* What is the difference?
  + How strongly we feel in our beliefs…as quantified by…

## More More Beta Distribution

The Beta distribution has a variance of

* Choosing and yields a prior with mean and variance
* Choosing and yields a prior with mean and variance
* The smaller prior variance means the prior is more informative
  + Informative priors are those that have relatively small variances
  + Uninformative priors are those that have relatively large variances

## Visualizing

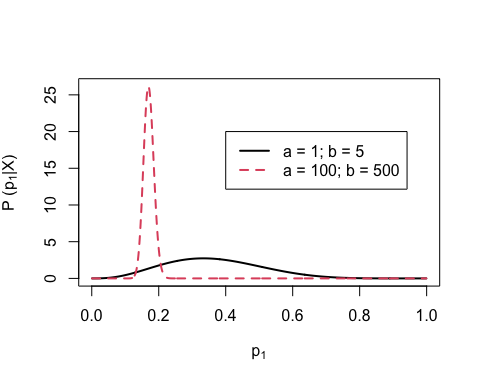


## The Posterior Distribution

Choosing a Beta distribution for a prior for is *very* convenient

* When combined with Bernoulli (Binomial) data likelihood the posterior distribution can be derived analytically
* The posterior distribution is also a Beta distribution
  + ( is the hyperparameter of the prior distribution)
  + ( is the hyperparameter of the posterior distribution)
* The Beta prior is said to be a conjugate prior: A prior distribution that leads to a posterior distribution of the same family
  + Here, prior == Beta and posterior == Beta

## Visualizing The Posterior Distribution



## Bayesian Estimates are Summaries of the Posterior Distribution

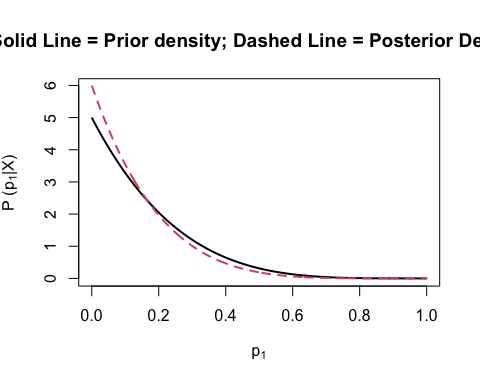
To determine the estimate of , we use summaries of the posterior distribution:

* With prior hyperparameters and
  + SD = 0.1388659
* With prior hyperparameters and
  + SD = 0.0152679
* The standard deviation (SD) of the posterior distribution is analogous to the standard error in frequentist statistics

## Bayesian Updating

We can use the posterior distribution as a prior!

Let’s roll a die to find out how…



## Section Summary

This section was a very quick introduction to Bayesian concepts:

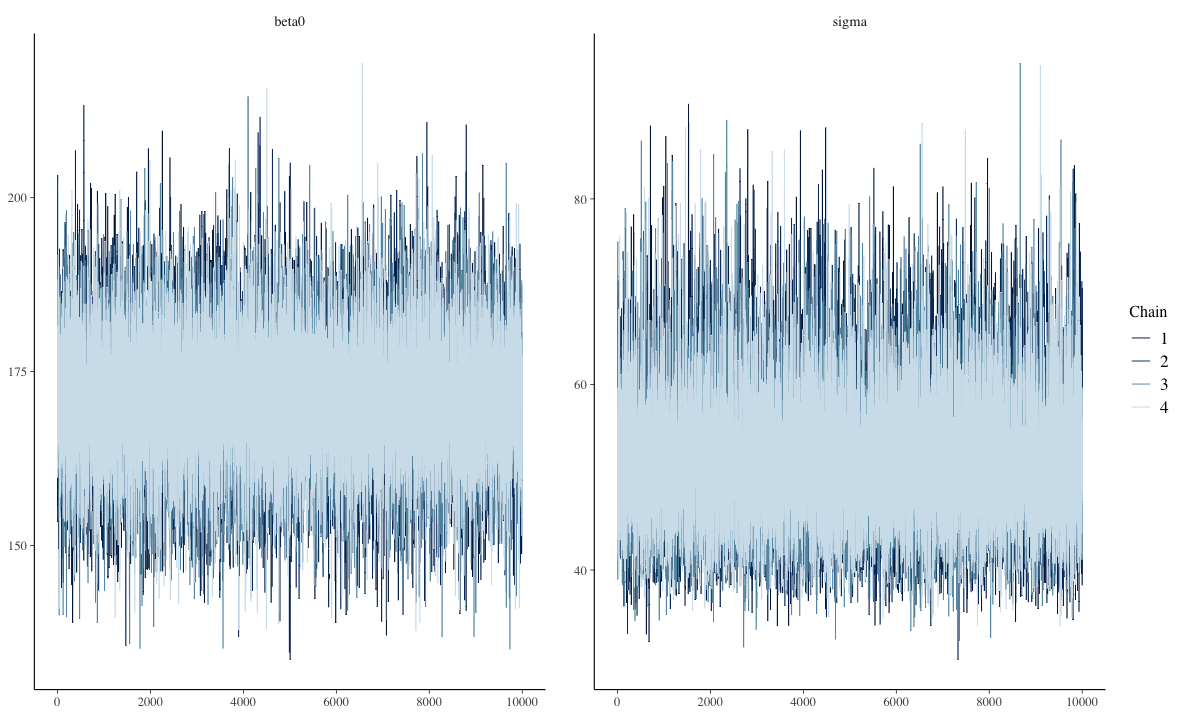
* prior distribution
  + hyperparameters
  + informative/uninformative
  + conjugate prior
* data likelihood
* posterior distribution
* Next we will discuss MCMC and Stan

# MCMC and Stan

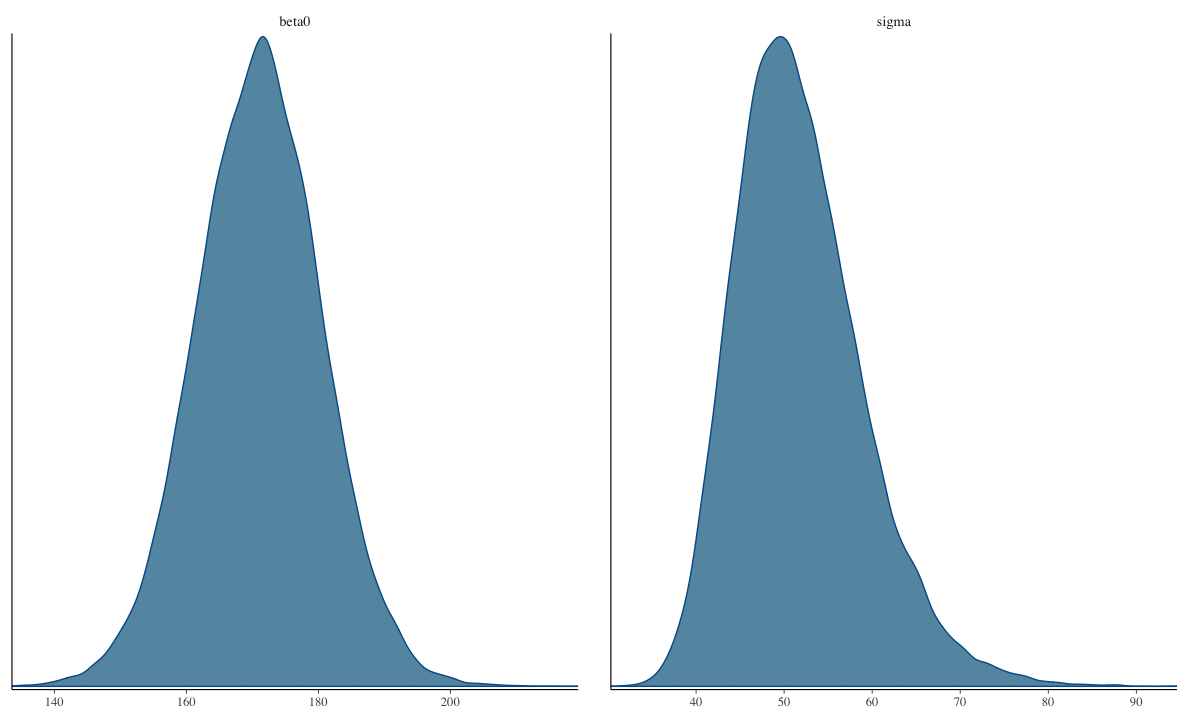
## Section Objectives

1. An Introduction to MCMC
2. An Introduction to Stan
3. Both with Linear Models

## The Markov Chain Timeseries



## The Posterior Distribution



## Markov Chain Monte Carlo Estimation

Bayesian analysis is all about estimating the posterior distribution

* Up until now, we have worked with posterior distributions that are fairly well-known
  + Beta-Binomial had a Beta distribution
  + In general, likelihood distributions from the exponential family have conjugate priors
    - Conjugate prior: the family of the prior is equivalent to the family of the posterior
* Most of the time, however, posterior distributions are not easily obtainable
  + No longer able to use properties of the distribution to estimate parameters

## Markov Chain Monte Carlo Estimation

* It is possible to use an optimization algorithm (e.g., Newton-Raphson or Expectation-Maximization) to find maximum value of posterior distribution
  + But, such algorithms may take a very long time for high-dimensional problems
* Instead: “sketch” the posterior by sampling from it – then use that sketch to make inferences
  + Sampling is done via MCMC

## Markov Chain Monte Carlo Estimation

* MCMC algorithms iteratively sample from the posterior distribution
  + For fairly simplistic models, each iteration has independent samples
  + Most models have some layers of dependency included
    - Can slow down sampling from the posterior
* There are numerous variations of MCMC algorithms
  + Most of these specific algorithms use one of two types of sampling:
    1. Direct sampling from the posterior distribution (i.e. Gibbs sampling)
       - Often used when conjugate priors are specified
    2. Indirect (rejection-based) sampling from the posterior distribution (e.g., Metropolis-Hastings, Hamiltonian Monte Carlo)

## MCMC Algorithms

* Efficiency is the main reason why there are many different algorithms
  + Efficiency in this context: How quickly the algorithm converges and provides adequate coverage (“sketching”) of the posterior distribution
  + No one algorithm is uniformly most efficient for all models (here model = likelihood prior)
* The good news is that many software packages (stan, JAGS, MPlus, especially) don’t make you choose which specific algorithm to use
* The bad news is that sometimes your model may take a large amount of time to reach convergence (think days or weeks)
* You can also code your own custom algorithm to make things run more smoothly

## Commonalities Across MCMC Algorithms

* Despite having fairly broad differences regarding how algorithms sample from the posterior distribution, there are quite a few things that are similar across algorithms:
  1. A period of the Markov chain where sampling is not directly from the posterior
     + The burnin period (sometimes coupled with other tuning periods and called warm-up)
  2. Methods used to assess convergence of the chain to the posterior distribution
     + Often involving the need to use multiple chains with independent and differing starting values

## Commonalities Across MCMC Algorithms

* Despite having fairly broad differences regarding how algorithms sample from the posterior distribution, there are quite a few things that are similar across algorithms:
  1. Summaries of the posterior distribution
* Further, rejection-based sampling algorithms often need a tuning period to make the sampling more efficient
  + The tuning period comes before the algorithm begins its burnin period

## MCMC Demonstration

* To demonstrate each type of algorithm, we will use a model for a normal distribution
  + We will investigate each, briefly
  + We will then switch over to stan to show the syntax and let stan work
  + We will conclude by talking about assessing convergence and how to report parameter estimates.

## Example Data: Post-Diet Weights

Example Data: <https://stats.idre.ucla.edu/spss/library/spss-libraryhow-do-i-handle-interactions-of-continuous-andcategorical-variables/>

* The file DietData.csv contains data from 30 respondents who participated in a study regarding the effectiveness of three types of diets.
* Variables in the data set are:
  1. Respondent: Respondent number 1-30
  2. DietGroup: A 1, 2, or 3 representing the group to which a respondent was assigned
  3. HeightIN: The respondent’s height in inches
  4. WeightLB (the Dependent Variable): The respondent’s weight, in pounds, recorded following the study

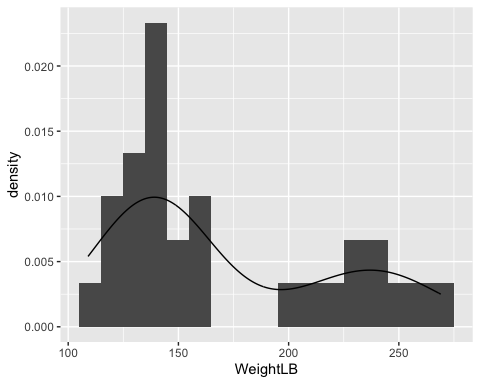
## Example Data: Post-Diet Weights

* The research question: Are there differences in final weights between the three diet groups, and, if so, what are the nature of the differences?
* But first, let’s look at the data

## Visualizing Data: WeightLB Variable

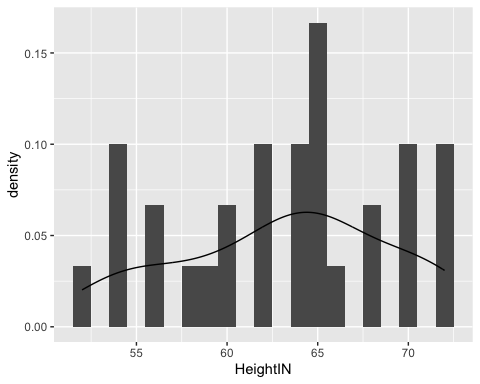
DietData = read.csv(file = "/Users/jonathantemplin/Library/CloudStorage/Dropbox/repos/mlmmWorkshop2023/dissemination/data/DietData.csv")  
  
ggplot(data = DietData, aes(x = WeightLB)) +   
 geom\_histogram(aes(y = ..density..), position = "identity", binwidth = 10) +   
 geom\_density(alpha=.2)

Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.  
ℹ Please use `after\_stat(density)` instead.



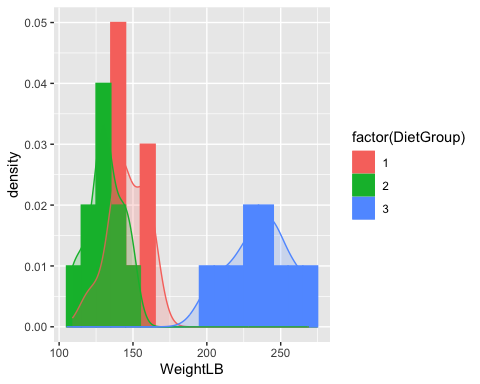
## Visualizing Data: HeightIN Variable

ggplot(data = DietData, aes(x = HeightIN)) +   
 geom\_histogram(aes(y = ..density..), position = "identity", binwidth = 1) +   
 geom\_density(alpha=.2)



## Visualizing Data: WeightLB by Group

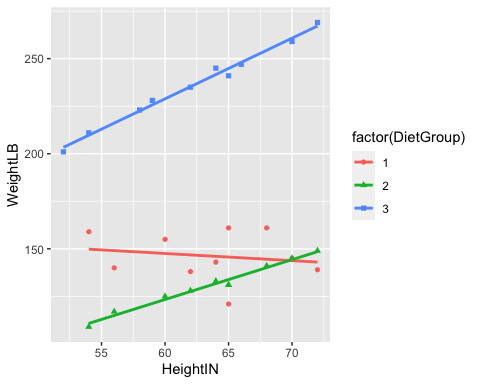
ggplot(data = DietData, aes(x = WeightLB, color = factor(DietGroup), fill = factor(DietGroup))) +   
 geom\_histogram(aes(y = ..density..), position = "identity", binwidth = 10) +   
 geom\_density(alpha=.2)



## Visualizing Data: Weight by Height by Group

ggplot(data = DietData, aes(x = HeightIN, y = WeightLB, shape = factor(DietGroup), color = factor(DietGroup))) +  
 geom\_smooth(method = "lm", se = FALSE) + geom\_point()

`geom\_smooth()` using formula = 'y ~ x'



## Class Discussion: What Do We Do?

Now, your turn to answer questions:

1. What type of analysis seems most appropriate for these data?
2. Is the dependent variable (WeightLB) is appropriate as-is for such analysis or does it need transformed?

## Linear Model with Least Squares

Let’s play with models for data…

# center predictors for reasonable numbers  
DietData$HeightIN60 = DietData$HeightIN-60  
  
# full analysis model suggested by data:  
FullModel = lm(formula = WeightLB ~ 1, data = DietData)  
  
# examining assumptions and leverage of fit  
# plot(FullModel)  
  
# looking at ANOVA table  
# anova(FullModel)  
   
# looking at parameter summary  
 summary(FullModel)

Call:  
lm(formula = WeightLB ~ 1, data = DietData)  
  
Residuals:  
 Min 1Q Median 3Q Max   
-62.00 -36.75 -24.00 49.00 98.00   
  
Coefficients:  
 Estimate Std. Error t value Pr(>|t|)   
(Intercept) 171.000 9.041 18.91 <2e-16 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 49.52 on 29 degrees of freedom

## Path Diagram of Our Model



## Steps in an MCMC Analysis

1. Specify model
2. Specify prior distributions for all model parameters
3. Build model syntax as needed
4. Run Markov chains (specify warmup/burnin and sampling period lengths)
5. Evaluate chain convergence
6. Interpret/report results

## Specify Model

* To begin, let’s start with an empty model and build up from there
* Let’s examine the linear model we seek to estimate:

Where:

Questions:

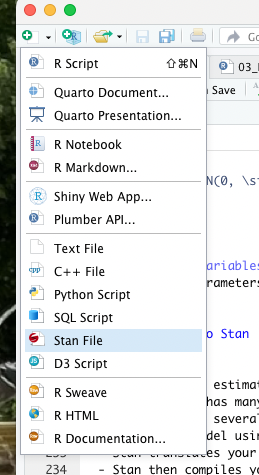
* What are the *variables* in this analysis?
* What are the parameters in this analysis?

## Introduction to Stan

* Stan is an MCMC estimation program
  + Most recent; has many convenient features
  + Actually does severaly methods of estimation (ML, Variational Bayes)
* You create a model using Stan’s syntax
  + Stan translates your model to a custom-built C++ syntax
  + Stan then compiles your model into its own executable program
* You then run the program to estimate your model
  + If you use R, the interface can be seamless

## Stan and RStudio

* Stan has its own syntax which can be built in stand-alone text files
  + Rstudio will let you create one of these files in the new file menu
  + Rstudio also has syntax highlighting in Stan files
    - This is very helpful to learn the syntax
* Stan syntax can also be built from R character strings
  + Which is helpful when running more than one model per analysis



## Stan Syntax

data {  
 int<lower=0> N;  
 vector[N] y;  
}  
  
parameters {  
 real beta0;  
 real<lower=0> sigma;  
}  
  
model {  
 beta0 ~ normal(0, 1000); // prior for beta0  
 sigma ~ uniform(0, 100000); // prior for sigma  
 y ~ normal(beta0, sigma); // model for observed data  
}

* Above is the syntax for our model
  + Each line ends with a semi colon
  + Comments are put in with //
* Three blocks of syntax needed
  + Data: What Stan expects you will send to it for the analysis (using R lists)
  + Parameters: Where you specify what the parameters of the model are
  + Model: Where you specify the distributions of the priors and data

## Stan Data and Parameter Delcaration

Like many compiled languages, Stan expects you to declare what type of data/parameters you are defining:

* int: Integer values (no decimals)
* real: Floating point numbers
* vector: A one-dimensional set of real valued numbers

Sometimes, additional definitions are provided giving the range of the variable (or restricting the set of starting values):

* real<lower=0> sigma;

See: <https://mc-stan.org/docs/reference-manual/data-types.html> for more information

## Stan Data and Prior Distributions

* In the model section, you define the distributions needed for the model and the priors
  + The left-hand side is either defined in data or parameters
    - y ~ normal(beta0, sigma); // model for observed data
    - sigma ~ uniform(0, 100000); // prior for sigma
  + The right-hand side is a distribution included in Stan
    - You can also define your own distributions

See: <https://mc-stan.org/docs/functions-reference/index.html> for more information

## From Stan Syntax to Compilation

# compile model -- this method is for stand-alone stan files (uses cmdstanr)  
model00.fromFile = cmdstan\_model(stan\_file = "model00.stan")  
  
# or this method using the string text in R  
model00.fromString = cmdstan\_model(stan\_file = write\_stan\_file(stanModel))

* Once you have your syntax, next you need to have Stan translate it into C++ and compile an executable
* This is where cmdstanr and rstan differ
  + cmdstanr wants you to compile first, then run the Markov chain
  + rstan conducts compilation (if needed) then runs the Markov chain

## Building Data for Stan

# build R list containing data for Stan: Must be named what "data" are listed in analysis  
stanData = list(  
 N = nrow(DietData),  
 y = DietData$WeightLB  
)  
  
# snippet of Stan syntax:  
stanSyntaxSnippet = "  
data {  
 int<lower=0> N;  
 vector[N] y;  
}  
"

* Stan needs the data you declared in your syntax to be able to run
* Within R, we can pass this data to Stan via a list object
* The entries in the list should correspond to the data portion of the Stan syntax
  + In the above syntax, we told Stan to expect a single integer named N and a vector named y
* The R list object is the same for cmdstanr and rstan

## Running Markov Chains in cmdstanr

# run MCMC chain (sample from posterior)  
model00.samples = model00.fromFile$sample(  
 data = stanData,  
 seed = 1,  
 chains = 4,  
 parallel\_chains = 4,  
 iter\_warmup = 10000,  
 iter\_sampling = 10000  
)

* With the compiled program and the data, the next step is to run the Markov Chain (a process sometimes called sampling as you are sampling from a posterior distribution)
* In cmdstanr, running the chain comes from the $sample() function that is a part of the compiled program object
* You must specify:
  + The data
  + The random number seed
  + The number of chains (and parallel chains)
  + The number of warmup iterations (more detail shortly)
  + The number of sampling iterations

## Running Markov Chains in rstan

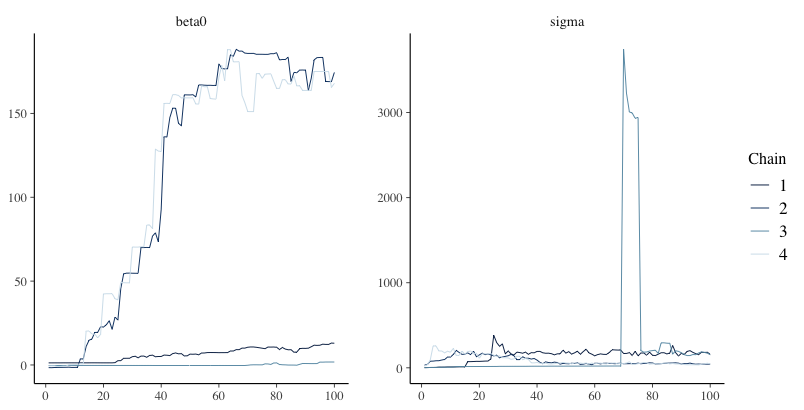
rstan\_options(auto\_write = TRUE)  
options(mc.cores = parallel::detectCores())  
  
# example MCMC analysis in rstan  
model00.rstan = stan(  
 model\_code = stanModel,  
 model\_name = "Empty model",  
 data = stanData,  
 warmup = 10000,  
 iter = 20000,  
 chains = 4,  
 verbose = TRUE  
)

* Rstan takes the model syntax directly, then compiles and runs the chains
* The first two lines of syntax enable running one chain per thread (parallel processing)
  + As chains are independent, running them simultaneously (parallel) shortens wait time considerably
* The verbose option is helpful for detecting when things break
* The same R list supplies the data to Stan

## MCMC Process

* The MCMC algorithm runs as a series of discrete iterations
  + Within each iteration, each parameter of a model has an opportunity to change its value
* For each parameter, a new parameter is sampled at random from the current belief of posterior distribution
  + The specifics of the sampling process differ by algorithm type (we’ll have a lecture on this later)
* In Stan (Hamiltonian Monte Carlo), for a given iteration, a proposed parameter is generated
  + The posterior likelihood “values” (more than just density; includes likelihood of proposal) are calculated for the current and proposed values of the parameter
  + The proposed values are accepted based on the draw of a uniform number compared to a transition probability
* If all models are specified correctly, then regardless of starting location, each chain will converge to the posterior if run long enough
  + But, the chains must be checked for convergence when the algorithm stops

## Example of Bad Convergence



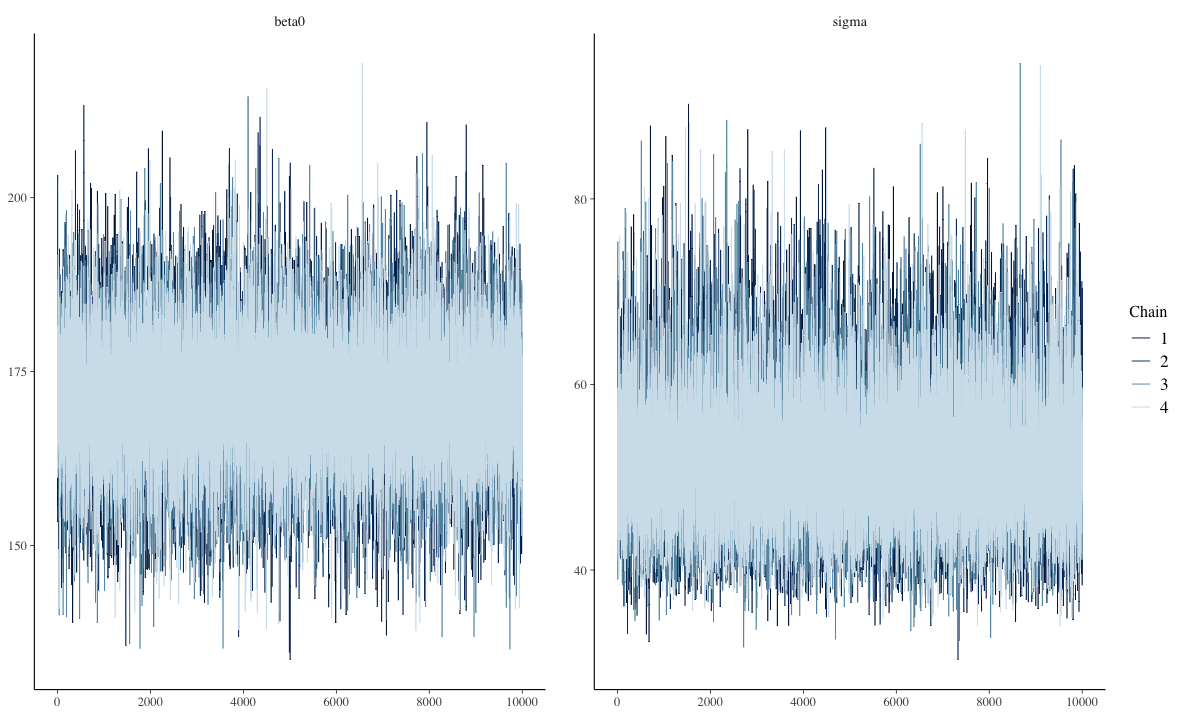
## Examining Chain Convergence

* Once Stan stops, the next step is to determine if the chains converged to their posterior distribution
  + Called convergence diagnosis
* Many methods have been developed for diagnosing if Markov chains have converged
  + Two most common: visual in spection and Gelman-Rubin Potential Scale Reduction Factor (PSRF; [quick reference](https://support.sas.com/documentation/cdl/en/statug/63962/HTML/default/viewer.htm#statug_introbayes_sect008.htm#statug.introbayes.bayesgelman))
* Visual inspection
  + Want no trends in timeseries – should look like a catapillar
  + Shape of posterior density should be mostly smooth
* Gelman-Rubin PSRF (denoted with )
  + For analyses with multiple chains
  + Ratio of between-chain variance to within-chain variance
  + Should be near 1 (maximum somewhere under 1.1)

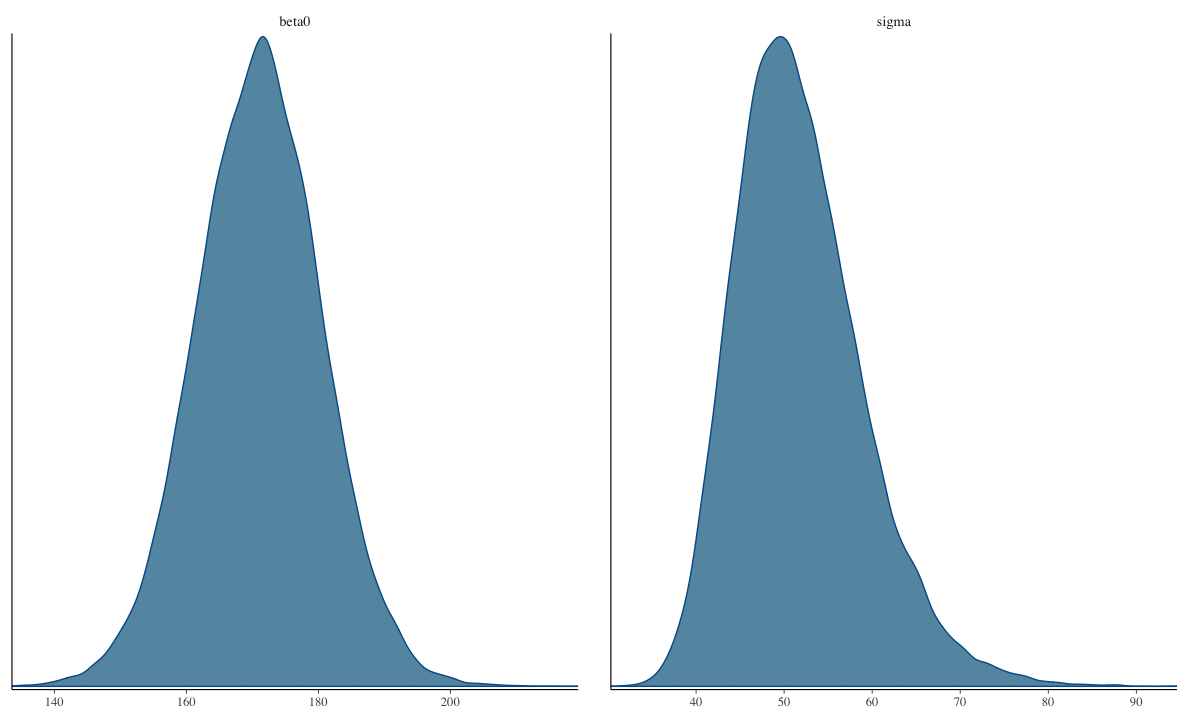
## Setting MCMC Options

* As convergence is assessed using multiple chains, more than one should be run
  + Between-chain variance estimates improve with the number of chains, so I typically use four
  + Others have two; more than one should work
* Warmup/burnin period should be long enough to ensure chains move to center of posterior distribution
  + Difficult to determine ahead of time
  + More complex models need more warmup/burnin to converge
* Sampling iterations should be long enough to thoroughly sample posterior distribution
  + Difficulty to determine ahead of time
  + Need smooth densities across bulk of posterior
* Often, multiple analyses (with different settings) are needed

## The Markov Chain Timeseries



## The Posterior Distribution



## Assessing Our Chains

model00.samples$summary()

# A tibble: 3 × 10  
 variable mean median sd mad q5 q95 rhat ess\_bulk ess\_tail  
 <chr> <num> <num> <num> <num> <num> <num> <num> <num> <num>  
1 lp\_\_ -129. -128. 1.03 0.737 -131. -128. 1.00 17204. 21369.  
2 beta0 171. 171. 9.52 9.40 155. 187. 1.00 26136. 23619.  
3 sigma 51.8 51.0 7.25 6.89 41.5 64.9 1.00 25343. 23597.

* The summary function reports the PSRF (rhat)
* Here we look at our two parameters: and
* Both have , so both would be considered converged
* lp\_\_ is posterior log likelihood–does not necessarily need examined
* ess\_ columns show effect sample size for chain (factoring in autocorrelation between correlations)
  + More is better

## Results Interpretation

* At long last, with a set of converged Markov chains, we can now interpret the results
  + Here, we disregard which chain samples came from and pool all sampled values to use for results
* We use summaries of posterior distributions when describing model parameters
  + Typical summary: the posterior mean
    - The mean of the sampled values in the chain
  + Called EAP (Expected a Posteriori) estimates
  + Less common: posterior median
* Important point:
  + Posterior means are different than what characterizes the ML estimates
    - Analogous to ML estimates would be the mode of the posterior distribution
  + Especially important if looking at non-symmetric posterior distributions
    - Look at posterior for variances

## Results Interpretation

* To summarize the uncertainty in parameters, the posterior standard deviation is used
  + The standard deviation of the sampled values in the chain
  + This is the analogous to the standard error from ML
* Bayesian credible intervals are formed by taking quantiles of the posterior distribution
  + Analogous to confidence intervals
  + Interpretation slightly different – the probability the parameter lies within the interval
  + 95% credible interval notes that parameter is within interval with 95% confidence
* Additionally, highest density posterior intervals can be formed
  + The narrowest range for an interval (for unimodal posterior distributions)

## Our Results

model00.samples$summary()

# A tibble: 3 × 10  
 variable mean median sd mad q5 q95 rhat ess\_bulk ess\_tail  
 <chr> <num> <num> <num> <num> <num> <num> <num> <num> <num>  
1 lp\_\_ -129. -128. 1.03 0.737 -131. -128. 1.00 17204. 21369.  
2 beta0 171. 171. 9.52 9.40 155. 187. 1.00 26136. 23619.  
3 sigma 51.8 51.0 7.25 6.89 41.5 64.9 1.00 25343. 23597.

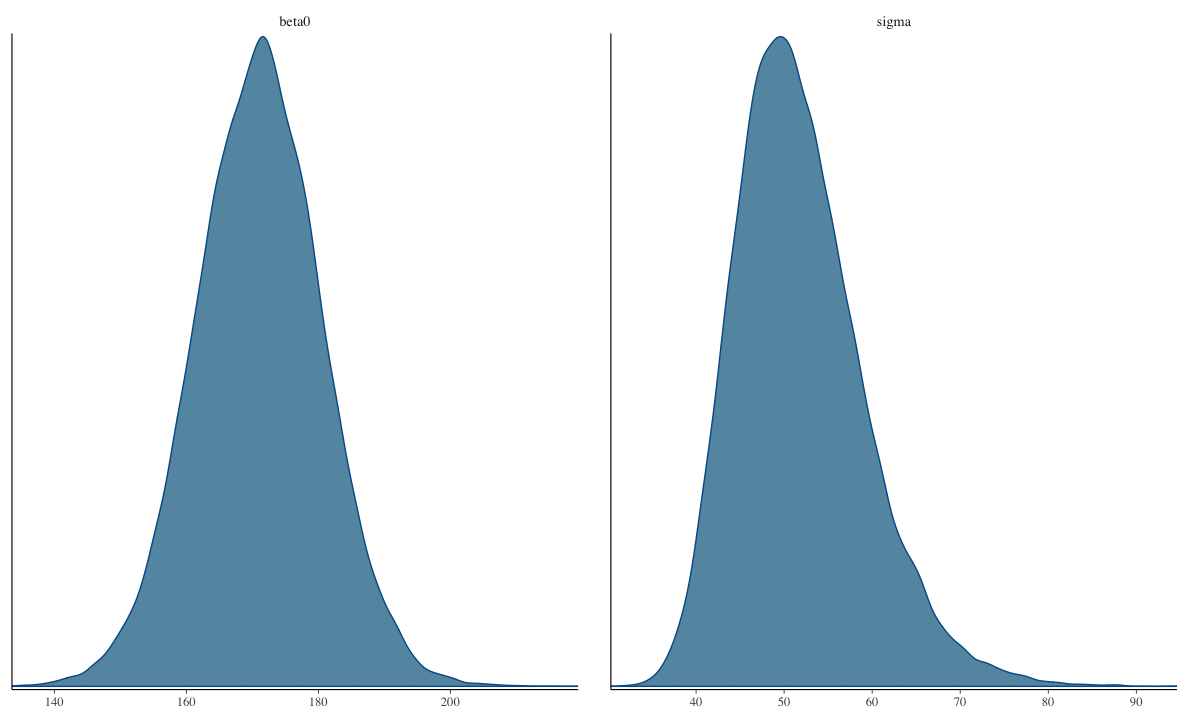
hdi(model00.samples$draws("beta0"), credMass = .9)

lower upper   
154.917 186.013   
attr(,"credMass")  
[1] 0.9

hdi(model00.samples$draws("sigma"), credMass = .9)

lower upper   
40.3082 63.1433   
attr(,"credMass")  
[1] 0.9

## The Posterior Distribution



## Wrapping Up

* This section covered the basics of MCMC estimation with Stan
* Next we will use an example to show a full analysis of the item response data problem we started with today
* The details today are the same for all MCMC analyses, regardless of which algorithm is used

# Estimating Bayesian IRT Models in Stan

## Section Objectives

1. Show how to estimate unidimensional latent variable models with dichotomous data\*
2. Show how to estimate different parameterizations of IRT/IFA models
3. Describe how to obtain IRT/IFA auxiliary statistics from Markov Chains
4. Show variations of various dichotomous-data models

* Also known as Item Repsonse Theory (IRT) or Item Factor Analysis (IFA) models

Dichotomous Data Distribution: Bernoulli

## The Bernoulli Distribution

The Bernoulli distribution is a one-trial version of the Binomial distribution

* Sample space (support)

The probability mass function (pdf):

The distribution has only one parameter: (the probability )

* Mean of the distribution:
* Variance of the distribution:

## Definition: Dichotomous vs. Binary

Note the definitions of some of the words for data with two values:

* Dichotomous: Taking two values (without numbers attached)
* Binary: either zero or one (specifically: )

Therefore:

* Not all dichotomous variables are binary (i.e., is a dichotomous variable)
* All binary variables are dichotomous

Finally:

* Bernoulli distributions are for binary variables
* Most dichotomous variables can be recoded as binary variables without loss of model effects

## Models with Bernoulli Distributions

Generalized linear models using Bernoulli distributions put a linear model onto a transformation of the mean

* Link functions map the mean from its original range of to

For an unconditional (empty) model, this is shown here:

## Link Functions for Bernoulli Distributions

Common choices for the link function (in latent variable models):

* Logit (or log odds):
* Probit:

Where Phi is the inverse cumulative distribution of a standard normal distribution:

## Less Common Link Functions

In the generalized linear models literature, there are a number of different link functions:

* Log-log:
* Complementary Log-Log:

Most of these seldom appear in latent variable models

* Each has a slightly different curve shape

## Inverse Link Functions

Our latent variable models will be defined on the scale of the link function

* Sometimes we wish to convert back to the scale of the data
  + Example: Test characteristic curves mapping onto an expected test score

For this, we need the inverse link function

* Logit (or log odds) link function:
* Logit (or log odds) inverse link function:

Latent Variable Models with Bernoulli Distributions for Observed Variables

## Latent Variable Models with Bernoulli Distributions for Observed Variables

We can finally define a latent variable model for binary responses using a Bernoulli distribution

* To start, we will use the logit link function
* We will begin with the linear predictor we had from the normal distribution models (Confirmatory factor analysis: )

For an item and a person , the model becomes:

* Note: the mean is replaced by
  + This is the mean of the observed variable, conditional on
* The item intercept is : The expected logit when
* The item discrimination is : The change in the logit for a one-unit increase in

## Model Family Names

Depending on your field, the model from the previous slide can be called:

* The two-parameter logistic (2PL) model with slope/intercept parameterization
* An item factor model

These names reflect the terms given to the model in diverging literatures:

* 2PL: Educational measurement
  + Birnbaum, A. (1968). Some Latent Trait Models and Their Use in Inferring an Examinee’s Ability. In F. M. Lord & M. R. Novick (Eds.), Statistical Theories of Mental Test Scores (pp. 397-424). Reading, MA: Addison-Wesley.
* Item factor analysis: Psychology
  + Christofferson, A.(1975). Factor analysis of dichotomous variables. Psychometrika , 40, 5-22.

Estimation methods are the largest difference between the two families

## Differences from Normal Distributions

Recall our normal distribution models:

Compared to our Bernoulli distribution models:

Differences:

* No residual (unique) variance in Bernoulli distribution
  + Only one parameter in distribution; variance is a function of the mean
* Identity link function in normal distribution:
  + Model scale and data scale are the same
* Logit link function in Bernoulli distribution
  + Model scale is different from data scale

## From Model Scale to Data Scale

Commonly, the IRT or IFA model is shown on the data scale (using the inverse link function):

The core of the model (the terms in the exponent on the right-hand side) is the same

* Models are equivalent
  + is on the data scale
  + is on the model (link) scale

## Modeling All Data

As with the normal distribution (CFA) models, we use the Bernoulli distribution for all observed variables:

## Measurement Model Analysis Steps

1. Specify model
2. Specify scale identification method for latent variables
3. Estimate model
4. Examine model-data fit
5. Iterate between steps 1-4 until adequate fit is achieved

#### Measurement Model Auxiliary Components

1. Score estimation (and secondary analyses with scores)
2. Item evaluation
3. Scale construction
4. Equating
5. Measurement invariance/differential item functioning

## Model Specification

The set of equations on the previous slide formed step #1 of the Measurement Model Analysis Steps:

1. Specify Model

The next step is:

1. Specify scale identification method for latent variables

We will initially assume , which allows us to estimate all item parameters of the model

* This is what we call a standardized latent variable
  + They are like Z-scores

## Identification of Latent Traits, Part 1

Psychometric models require two types of identification to be valid:

1. Empirical Identification

* The minimum number of items that must measure each latent variable
* From CFA: three observed variables for each latent variable (or two if the latent variable is correlated with another latent variable)

Bayesian priors can help to make models with fewer items than these criteria suggest estimable

* The parameter estimates (item parameters and latent variable estimates) often have MCMC convergence issues and should not be trusted
* Use the CFA standard in your work

## Identification of Latent Traits, Part 2

Psychometric models require two types of identification to be valid:

1. Scale Identification (i.e., what the mean/variance is for each latent variable)

* The additional set of constraints needed to set the mean and standard deviation (variance) of the latent variables
* Two main methods to set the scale:
  + Marker item parameters
    - For variances: Set the loading/slope to one for one observed variable per latent variable
      * Can estimate the latent variable’s variance (the diagonal of )
    - For means: Set the item intercept to one for one observed variable perlatent variable
      * Can estimate the latent variable’s mean (in )
  + Standardized factors
    - Set the variance for all latent variables to one
    - Set the mean for all latent variables to zero
    - Estimate all unique off-diagonal correlations (covariances) in

## More on Scale Identification

Bayesian priors can let you believe you can estimate more parameters than the non-Bayesian standards suggest

* For instance, all item parameters and the latent variable means/variances

Like empirical identification, these estimates are often unstable and are not recommended

Most common:

* Standardized latent variables
  + Used for scale development and/or when scores are of interest directly
* Marker item for latent variables and zero means
  + Used for cases where latent variables may become outcomes (and that variance needs explained)

#### Important Point: Regardless of model choice, model/data likelihoods are equivalent

* Differing prior distributions may make models non-equivalent

## Model (Data) Likelihood Functions

The specification of the model defines the model (data) likelihood function for each type of parameter

* To demonstrate, let’s examine the data likelihood for the factor loading for the first item

The model (data) likelihood function can be defined conditional on all other parameter values (as in a block in an MCMC iteration)

* That is: hold and constant

The likelihood is then:

## Model (Data) Log Likelihood Functions

As this number can be very small (making numerical precision an issue), we often take the log:

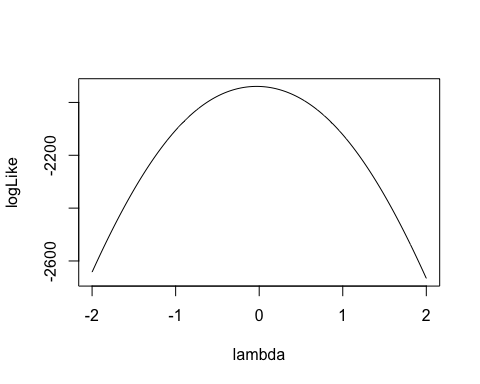
The key in the likelihood function is to substitute each person’s data-scale model for :

Which then becomes:

## Model (Data) Log Likelihood Functions

As an example for :

load("data/modelingData.RData")  
  
mu1 = -2  
theta = rnorm(n = nrow(modelingData), mean = 0, sd = 1)  
  
lambda = seq(-2,2, .01)  
logLike = NULL  
  
param=1 # for demonstrating  
for (param in 1:length(lambda)){  
   
 logit = mu1 + lambda[param]\*theta  
 prob = exp(logit)/(1+exp(logit))  
 bernoulliLL = sum(dbinom(x = modelingData$score1, size = 1, prob = prob, log = TRUE))  
   
 logLike = c(logLike, bernoulliLL)  
}  
  
plot(x = lambda, y = logLike, type = "l")



## Model (Data) Log Likelihood Functions for

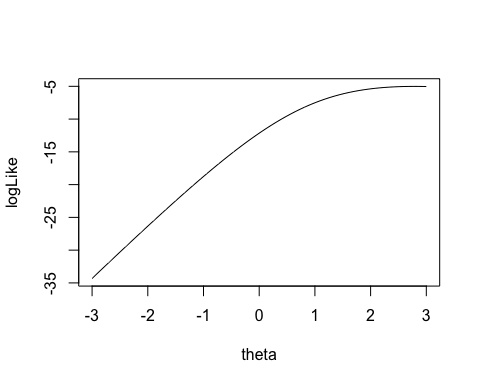
For each person, the same model (data) likelihood function is used

* Only now it varies across each item response
* Example: Person 1

## Model (Data) Log Likelihood Functions

As an example for the log-likelihood for :

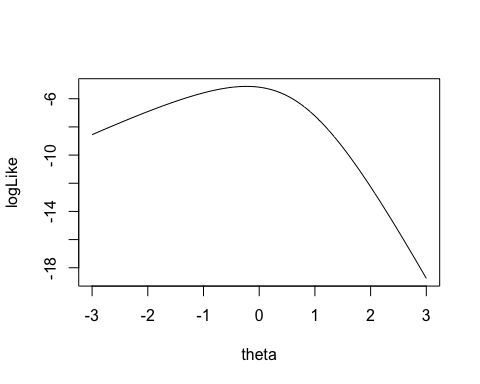
responseData = modelingData[paste0("score",1:10)]  
  
person = 2  
  
# for theta2  
mu = runif(n = ncol(responseData), min = -2, max = 0)  
lambda = runif(n = ncol(responseData), min = 0, max = 2)  
  
  
theta = seq(-3,3,.01)  
logLike = NULL  
  
param=1 # for demonstrating  
for (param in 1:length(theta)){  
 thetaLL = 0  
 for (item in 1:ncol(responseData)){  
 logit = mu[item] + lambda[item]\*theta[param]  
 prob = exp(logit)/(1+exp(logit))  
 thetaLL = thetaLL + dbinom(x = responseData[person,item], size = 1, prob = prob, log = TRUE)  
 }  
   
 logLike = c(logLike, thetaLL)  
}  
  
plot(x = theta, y = logLike, type = "l")



## Model (Data) Log Likelihood Functions

As an example for the log-likelihood for :

person = 1  
  
# for theta2  
mu = runif(n = ncol(responseData), min = -2, max = 0)  
lambda = runif(n = ncol(responseData), min = 0, max = 2)  
  
  
  
theta = seq(-3,3,.01)  
logLike = NULL  
  
param=1 # for demonstrating  
for (param in 1:length(theta)){  
 thetaLL = 0  
 for (item in 1:ncol(responseData)){  
 logit = mu[item] + lambda[item]\*theta[param]  
 prob = exp(logit)/(1+exp(logit))  
 thetaLL = thetaLL + dbinom(x = responseData[person,item], size = 1, prob = prob, log = TRUE)  
 }  
   
 logLike = c(logLike, thetaLL)  
}  
  
plot(x = theta, y = logLike, type = "l")



Implementing Bernoulli Outcomes in Stan

## Stan’s model Block

model {  
  
 lambda ~ multi\_normal(priorMeanLambda, priorCovLambda); // Prior for item discrimination/factor loadings  
 mu ~ multi\_normal(priorMeanMu, priorCovMu); // Prior for item intercepts  
   
 theta ~ normal(0, 1); // Prior for latent variable (with mean/sd specified)  
   
 for (item in 1:nItems){  
 Y[item] ~ bernoulli\_logit(mu[item] + lambda[item]\*theta);  
 }  
   
}

For logit models without lower/upper asymptote parameters, Stan has a convenient bernoulli\_logit() function

* Automatically has the link function embedded
* The catch: The data have to be defined as an integer

Also, note that there are few differences from the normal outcomes models (CFA)

* No residual variance parameters

## Stan’s parameters Block

parameters {  
 vector[nObs] theta; // the latent variables (one for each person)  
 vector[nItems] mu; // the item intercepts (one for each item)  
 vector[nItems] lambda; // the factor loadings/item discriminations (one for each item)  
}

## Stan’s data {} Block

data {  
 int<lower=0> nObs; // number of observations  
 int<lower=0> nItems; // number of items  
 array[nItems, nObs] int<lower=0, upper=1> Y; // item responses in an array  
  
 vector[nItems] priorMeanMu; // prior mean vector for intercept parameters  
 matrix[nItems, nItems] priorCovMu; // prior covariance matrix for intercept parameters  
  
 vector[nItems] priorMeanLambda; // prior mean vector for loading parameters  
 matrix[nItems, nItems] priorCovLambda; // prior covariance matrix for loading parameters  
}

One difference from normal outcomes model—the data are defined as an array:

array[nItems, nObs] int<lower=0, upper=1> Y;

* Arrays are types of matrices (with more than two dimensions possible)
  + Allows for different types of data (here Y are integers)
    - Integer-valued variables needed for bernoulli\_logit() function
* Arrays are row-major (meaning order of items and persons is switched)

## Change to Data List for Stan Import

The switch of items and observations in the array statement means the data imported have to be transposed:

# build stan data file  
 model2PL\_Data = list(  
 nObs = nrow(modelingData),  
 nItems = 10,  
 Y = t(correctResponseData),  
 priorMeanMu = rep(0, nItems),  
 priorCovMu = 10 \* diag(nItems),  
 priorMeanLambda = rep(0, nItems),  
 priorCovLambda = 10 \* diag(nItems)  
 )

## Running the Model In Stan

The Stan program takes longer to run than in linear models:

* Note: Typically, longer chains are needed for larger models like this
* Note: Starting values added (mean of 5 is due to logit function limits)
  + Helps keep definition of parameters (stay away from opposite mode)
  + Too large of value can lead to NaN values (exceeding numerical precision)

See 01\_unidimensionalIRT.R for syntax and results

## Modeling Strategy vs. Didactic Strategy

At this point, one should investigate model fit of the model we just ran

* If the model does not fit, then all model parameters could be biased
  + Both item parameters and person parameters ()
* Moreover, the uncertainty accompanying each parameter (the posterior standard deviation) may also be biased
  + Especially bad for psychometric models as we quantify reliaiblity with these numbers

But, to teach generalized measurement models, we will first talk about differing models for observed data

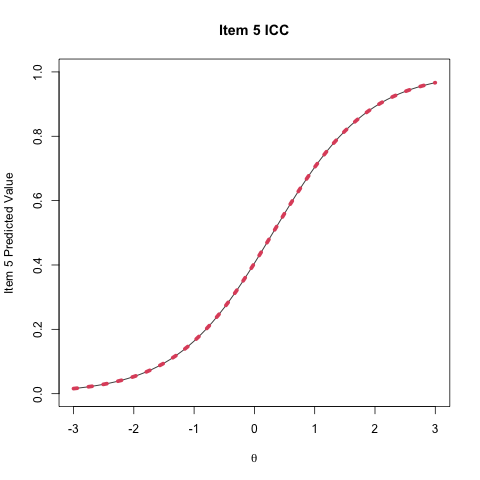
* Different distributions
* Different parameterizations across the different distributions

## Investigating Item Parameters

One plot that can help provide information about the item parameters is the item characteristic curve (ICC)

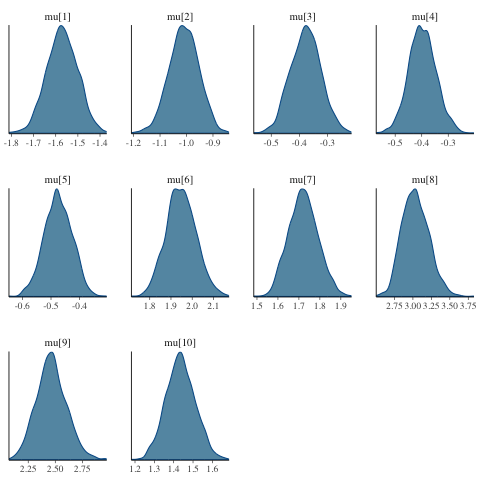
* The ICC is the plot of the expected value of the response conditional on the value of the latent traits, for a range of latent trait values
* Because we have sampled values for each parameter, we can plot one ICC for each posterior draw

## Posterior ICC Plots



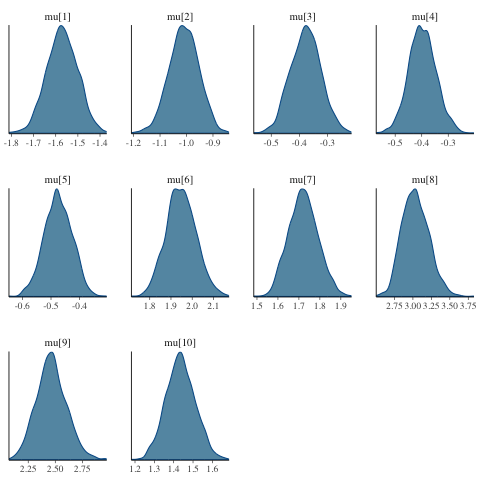
## Investigating the Item Parameters

Trace plots for



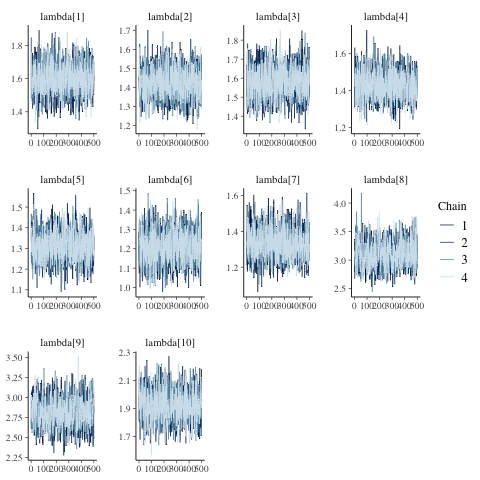
## Investigating the Item Parameters

Density plots for



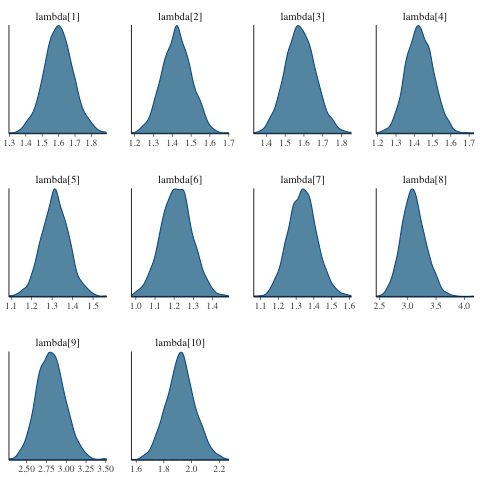
## Investigating the Item Parameters

Trace plots for



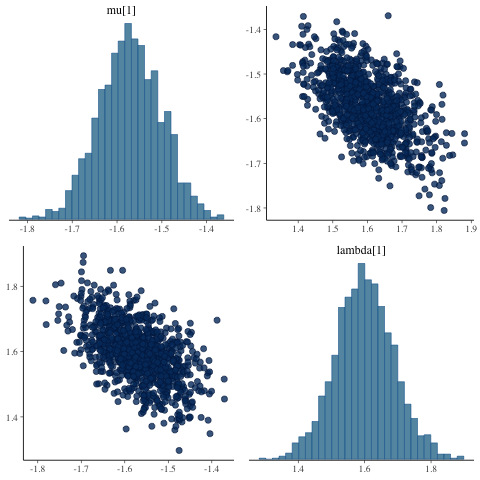
## Investigating the Item Parameters

Density plots for



## Investigating the Item Parameters

Bivariate plots for and



## Investigating the Latent Variables

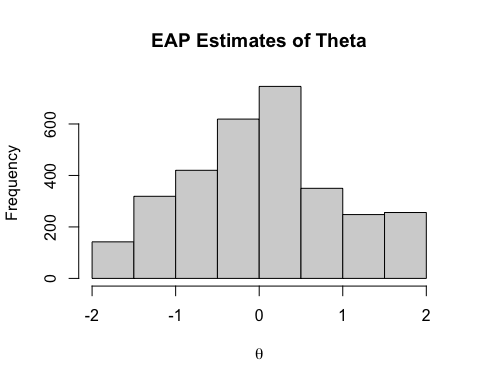
The estimated latent variables are then:

load("data/01\_model02.RData")  
print(model2PL\_Summary[grep(pattern = "theta", x=model2PL\_Summary$variable),])

# A tibble: 3,100 × 10  
 variable mean median sd mad q5 q95 rhat ess\_bulk ess\_tail  
 <chr> <num> <num> <num> <num> <num> <num> <num> <num> <num>  
 1 theta[1] -1.65 -1.62 0.481 0.479 -2.49 -0.895 0.999 4198. 1511.  
 2 theta[2] 0.772 0.746 0.493 0.495 -0.00466 1.60 1.00 5030. 1361.  
 3 theta[3] 0.263 0.246 0.445 0.461 -0.473 0.990 0.999 5262. 1613.  
 4 theta[4] -0.169 -0.183 0.414 0.406 -0.820 0.520 1.01 4806. 1068.  
 5 theta[5] -0.476 -0.489 0.380 0.366 -1.10 0.177 1.01 4162. 1280.  
 6 theta[6] -0.698 -0.707 0.387 0.375 -1.34 -0.0376 1.00 3513. 1210.  
 7 theta[7] -1.56 -1.52 0.461 0.444 -2.39 -0.867 1.00 3720. 1025.  
 8 theta[8] -0.391 -0.391 0.390 0.399 -1.01 0.263 1.00 4712. 1302.  
 9 theta[9] 0.702 0.689 0.484 0.490 -0.0386 1.55 1.01 4324. 1252.  
10 theta[10] -0.662 -0.662 0.386 0.386 -1.28 -0.0345 1.01 4424. 1302.  
# ℹ 3,090 more rows

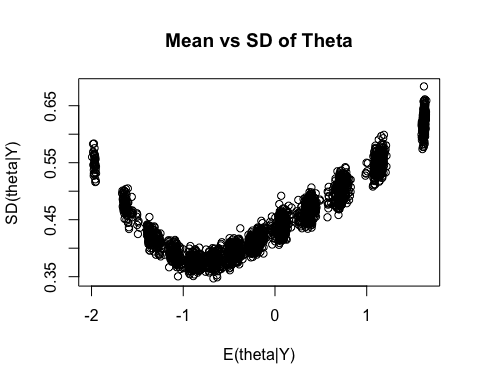
## EAP Estimates of Latent Variables

hist(  
 model2PL\_Summary$mean[grep(pattern = "theta", x=model2PL\_Summary$variable)], main="EAP Estimates of Theta", xlab = expression(theta)  
)



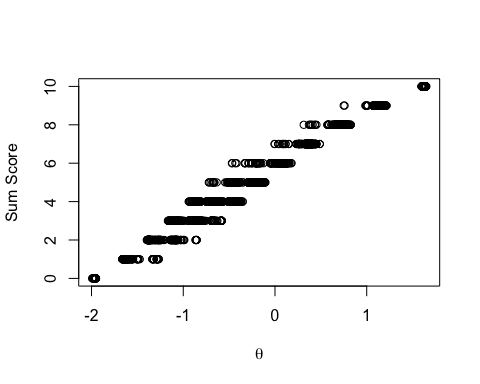
## Comparing EAP Estimates with Posterior SDs

plot(y = model2PL\_Summary$sd[grep(pattern = "theta", x=model2PL\_Summary$variable)],   
 x = model2PL\_Summary$mean[grep(pattern = "theta", x=model2PL\_Summary$variable)],  
 xlab = "E(theta|Y)", ylab = "SD(theta|Y)", main="Mean vs SD of Theta")



## Comparing EAP Estimates with Sum Scores

plot(y = modelingData$sumScore, x = model2PL\_Summary$mean[grep(pattern = "theta", x=model2PL\_Summary$variable)],  
 ylab = "Sum Score", xlab = expression(theta))



## Additional IRT Information

There is a lot more to IRT models, but most is tangiential to our workshop

* See Bayesian Psychometric Models my course notes at
  + 2022 (with Stan): <https://jonathantemplin.com/bayesian-psychometric-modeling-fall-2022/>
  + 2019 (with JAGS):<https://jonathantemplin.com/bayesian-psychometric-modeling-spring-2019/>

# Bayesian Psychometric Model Fit Methods

## Section Objectives

1. Show how to use PPMC to evaluate absolute model fit in Bayesian psychometric models
2. Show how to use LOO and WAIC for relative model fit in Bayesian psychometric models

Posterior Predictive Model Checking for Absolute Fit in Bayesian Psychometric Models

## Psychometric Model PPMC

Psychometric models can use posterior predictive model checking (PPMC) to assess how well they fit the data in an absolute sense

* At each iteration of the chain, each item is simulated using model parameter values from that iteration
* Summary statistics are used to evaluate model fit
  + Univariate measures (each item, separately):
    - Item mean
    - Item (comparing observed data with each simulated data set)
  + Not very useful unless:
    - Parameters have “obscenely” informative priors
    - There are cross-item constraints on some parameters (such as all loadings are equal)
  + Bivariate measures (each pair of items)
    - For binary data: Tetrachoric correlations
    - For polytomous data: Polychoric correlations (but difficult to estimate with small samples), pearson correlations
    - For other types of data: Pearson correlations

## Problems with PPMC

Problems with PPMC include

* No uniform standard for which statistics to use
  + Tetrachoric correlations? Pearson correlations?
* No uniform standard by which data should fit, absolutely
  + Jihong Zhang has some work on this topic, though:
    - Paper in [*Structural Equation Modeling*](https://www.tandfonline.com/doi/abs/10.1080/10705511.2021.2012682)
    - Dissertation on PPMC with M2 statistics (working on publishing)
* No way to determine if a model is overparameterized (too complicated)
  + Fit only improves to a limit

## Implementing PPMC in Stan (one )

generated quantities{  
  
 // for PPMC:  
 array[nItems, nObs] int<lower=0> simY;  
   
 for (item in 1:nItems){  
 for (obs in 1:nObs){  
 // generate data based on distribution and model  
 simY[item, obs] = bernoulli\_logit\_rng(mu[item] + lambda[item]\*theta[obs]);  
   
 }  
 }  
}

Notes:

* Generated quantities block is where to implement PPMC
* Each type of distribution also has a random number generator
  + Here, bernoulli\_logit\_rng goes with bernoulli\_logit
* Each may have some issue in types of inputs (had to go person-by-person in this block)
* Rather than have Stan calculate statistics, I will do so in R

## PPMC Processing

Stan generated a lot of data—but now we must take it from the format of Stan and process it:

* For this, we refer to the file 01\_unidimensionalIRT.R
* Each IRT (or ML IRT) model has a section for PPMC
* “Helper” function at top of syntax to calculate tetrachoric correlation between items

# Relative Model Fit in Bayesian Psychometric Models

## Relative Model Fit in Bayesian Psychometric Models

As with other Bayesian models, we can use WAIC and LOO to compare the model fit of two Bayesian models

* Of note: There is some debate as to whether or not we should marginalize across the latent variables
  + We won’t do that here as that would involve a numeric integral
* What is needed: The conditional log likelihood for each observation at each step of the chain
  + Here, we have to sum the log likelihood across all items
  + There are built-in functions in Stan to do this
* Each IRT (or ML IRT) model has a section for LOO/WAIC

## Implementing ELPD in Stan (one )

generated quantities{  
   
 // for LOO/WAIC:  
 vector[nObs] personLike = rep\_vector(0.0, nObs);  
   
 for (item in 1:nItems){  
 for (obs in 1:nObs){  
 // calculate conditional data likelihood for LOO/WAIC  
 personLike[obs] =   
 personLike[obs] +   
 bernoulli\_logit\_lpmf(Y[item, obs] | mu[item] + lambda[item]\*theta);  
 }  
 }  
}

Notes:

* bernoulli\_logit\_lpmf needs the observed data to work (first argument)
* vector[nObs] personLike = rep\_vector(0.0, nObs); is needed to set the values to zero at each iteration prior to summing

## Section Summary

Model fit is complicated for psychometric models

* Bayesian model fit methods are even more complicated than non-Bayesian methods
* Open area for research!

1. “Add”Stuff” Up” model: Ask Lesa what “stuff” means here…Scheiße zusammenzählen [↑](#footnote-ref-43)