Bayesian Statistics for College Juniors and Seniors

Note

There will be some pre-written code that you can either code yourself, or run what I've already written. Either way, we're going to use two R packages:

```
install.packages("rstan")
install.packages("alr3")
```

RStan can take several minutes to install! If you'd like to participate with us, you might want to install it now!

All code is posted in the following repository as well:

https://github.com/timbook/uscots-bayes-workshop

The Setting

College STEM majors typically take a **mathematical statistics** course in their junior or senior year.

This course is typically **calculus-based** and focuses on statistical distributions, random variables, and probability rules.

We imagine Bayesian statistics fitting neatly into **two or three weeks** of a typical fifteen-week semester course.

Sample Syllabus of Bayesian Statistics

Week One

- Recap of prerequisites
- Tour of named distributions
- Finding posterior distributions by hand
- Conjugacy
- Begin discussing non-conjugacy and the need for technological intervention

Programming Intermezzo

Considering the direction in which this field is moving, the topic of statistical programming cannot be avoided.

Prior student experience will probably be a mix of R, Matlab, and Python, but it's likely most students will probably have no programming experience whatsoever.

Given a three-week curriculum, the second week should be entirely devoted to programming fundamentals in relation to statistics. **We recommend R**, as it has the lowest barrier to entry, and it has the opportunity to tie in very nicely with various other materials learned in the course up until this point.

Sample Syllabus of Bayesian Statistics

Week Two (or Three?)

- MCMC fundamentals
- MCMC in R
- A one-layer hierarchical model
- Bayesian regression

Prereqs

Because of the nature of the course, much of the setup to learning Bayesian statistics is likely covered much earlier in the semester. Namely:

- Introduction of the usual “named” probability distributions
- Basic probability theory with random variables
- Conditional probability and hierarchical distributions

These materials should be recapped before beginning a discussion of “modern” Bayesian approaches. We believe the above topics can exhaustively be relearned in the context of discovering **conjugacy**.

Let’s Begin!

Week 1

Popular Distributions

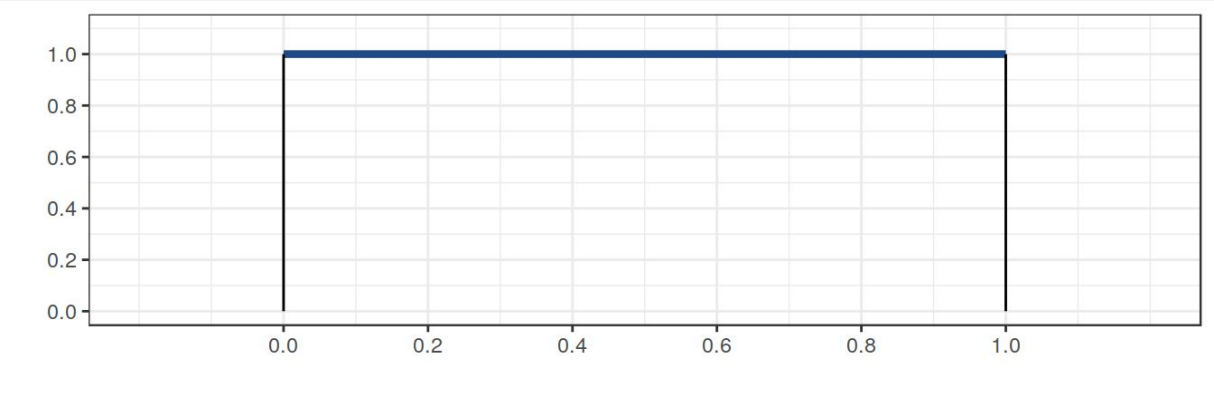
In order to be effective Bayesians, we'll need to have a wide array of distributions at our disposal.

Here are a few we've already covered in this course and a few additional useful ones.

The Uniform Distribution

The uniform distribution assigns equal probability between its two bounds *a* and *b*. Can be either continuous or discrete.

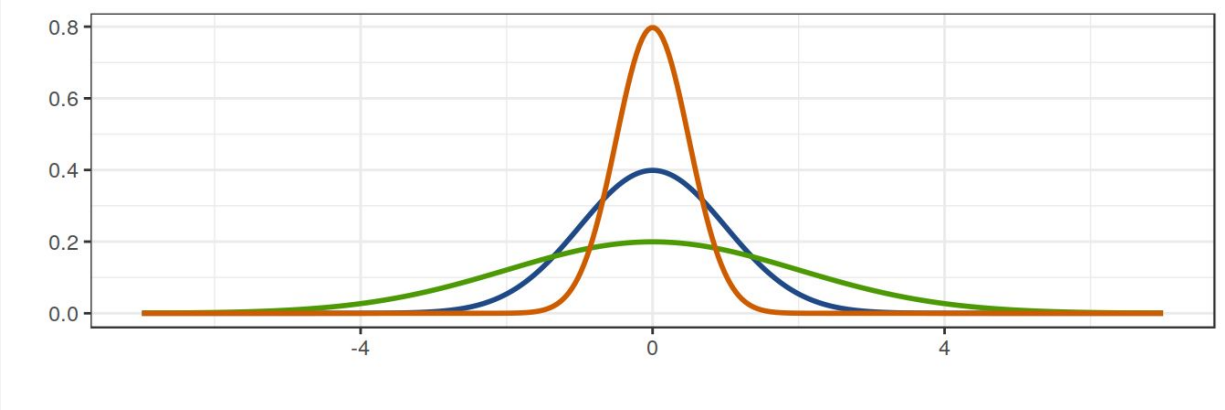
$$f(x) = \frac{1}{b-a}$$



The Normal Distribution

Found often in nature, the normal distribution is symmetric and bell-shaped. It concentrates its probability around its mean μ, and has standard deviation σ.

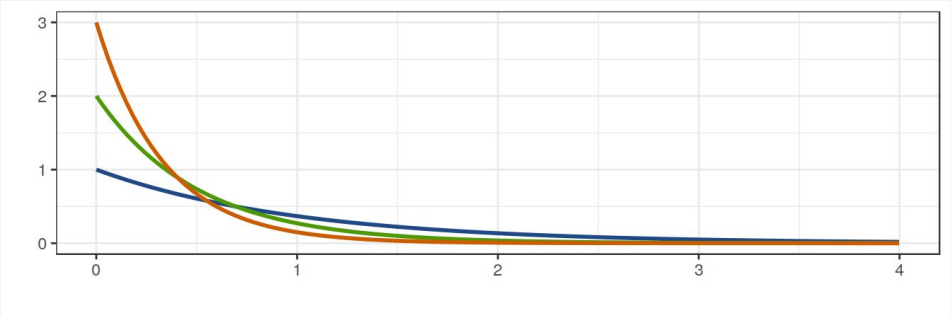
$$f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}}$$



The Exponential Distribution

The exponential distribution describes **waiting-time** distributions. For example, the time a light bulb lasts before going out might be exponentially distributed. It can be parametrized in terms of its rate λ (“light bulb lasts three years”) or its scale β (“⅓ of a light bulb per year”).

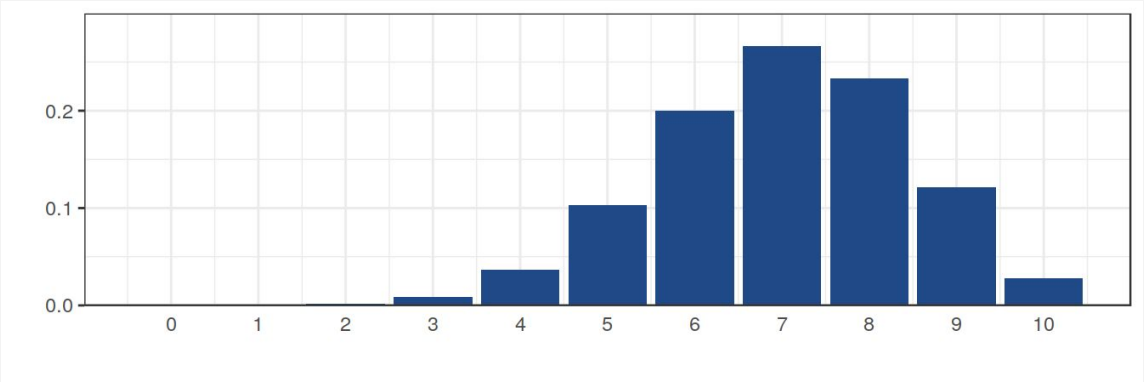
$$f(x) = \lambda e^{-\lambda x} = \frac{1}{\beta} e^{-x/\beta}$$



The Binomial Distribution

The binomial distribution describes the number of successes when conducting a trial with *n* successes, with fixed probability *p* of success. This is known as the **Bernoulli distribution** when *n* = 1.

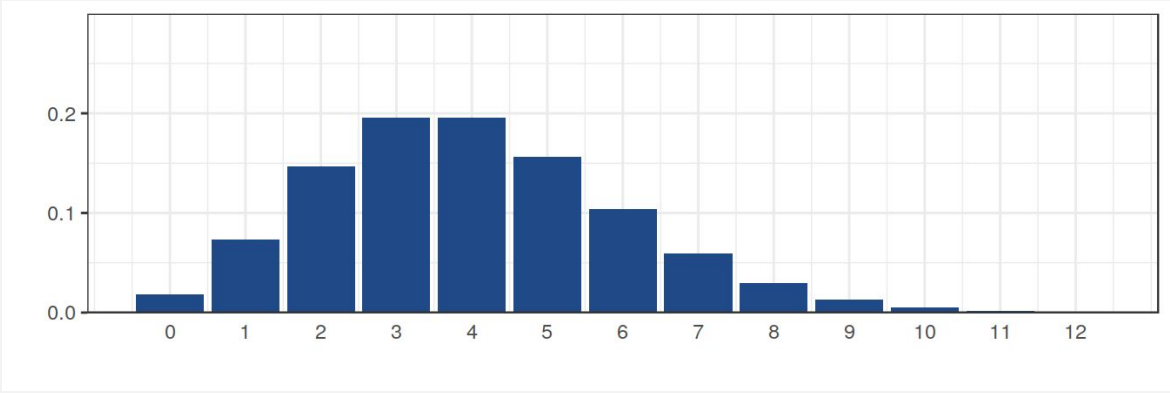
$$p(x) = \binom{n}{x} p^x (1-p)^{n-x}$$



The Poisson Distribution

The Poisson distribution describes the number of events in a fixed interval of time, with mean λ. For example, the number of customers that Corner Room serves during their lunch rush every day might be Poisson distributed.

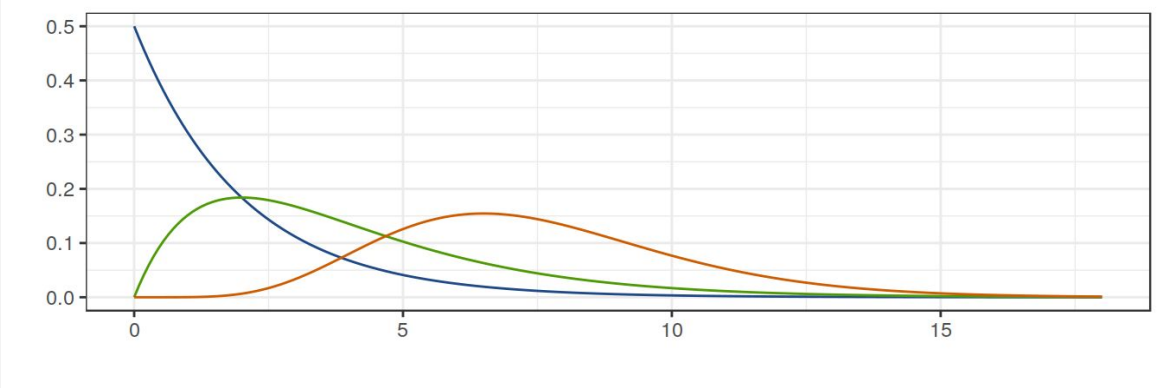
$$p(x) = \frac{e^{-\lambda}\lambda^{x}}{x!}$$



The Gamma Distribution

The gamma distribution is very similar in shape to the exponential distribution. It describes how long it takes for multiple events to occur. That is, it is the sum of multiple exponential distributions.

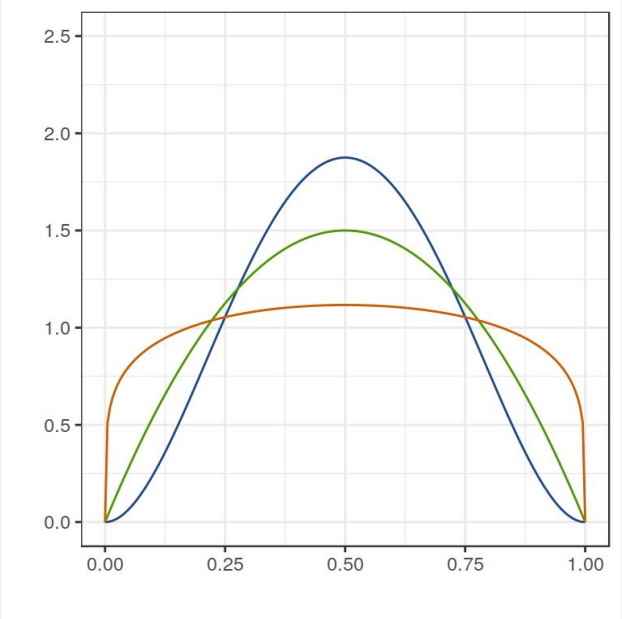
$$f(x) = cx^{\alpha-1}e^{-x/\beta}$$



The Beta Distribution

The beta distribution is defined over the interval [0, 1]. It is often used in Bayesian statistics to describe uncertainty within probabilities themselves.

$$f(x) = cx^{\alpha-1}(1-x)^{\beta-1}$$



Bayes' Theorem

Recall that Bayes' Theorem

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

can be applied to probability densities as well:

$$f(\theta|x) = \frac{f(x|\theta)f(\theta)}{f(x)}$$

But… what are f(θ) and f($\theta|x$)?!

How do fixed parameters have distributions, you might ask? In order to achieve full Bayesian zen, we must **rethink what probability means.**

A **frequentist** believes that probability is the chance that something occurs.

A **Bayesian** believes that probability reflects **their degree of belief that something will occur**.

Why is this distinction important? Let's take a look at an example.

Presidential Election

Who will win the presidency in 2020, Candidate A or B?

A **frequentist** will attempt to use polling data to find the probability a candidate will win, but are they correct? How can we know? **Elections are only held once!**

A **Bayesian** can incorporate **prior beliefs** alongside polling data. In the US, presidential elections almost always end up being close to 50/50. This is a good assumption going into an election. It is even used by famous Bayesian Nate Silver at FiveThirtyEight.com!

Presidential Election

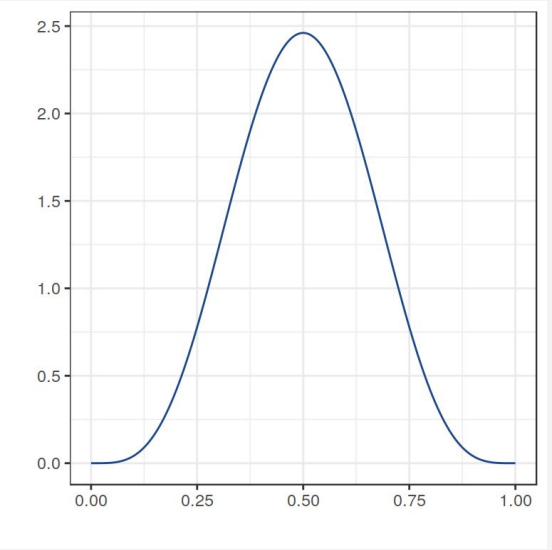
Candidate A will win the presidency in 2020 with probability p. I don't know p, but it is probably close to 50/50. Let X be an indicator of whether or not polls indicate that Candidate A will win. Let's consider the following setup:

$$X|p \sim \text{Bernoulli}(p)$$

$$p \sim \text{Beta}(5, 5)$$

prior belief!

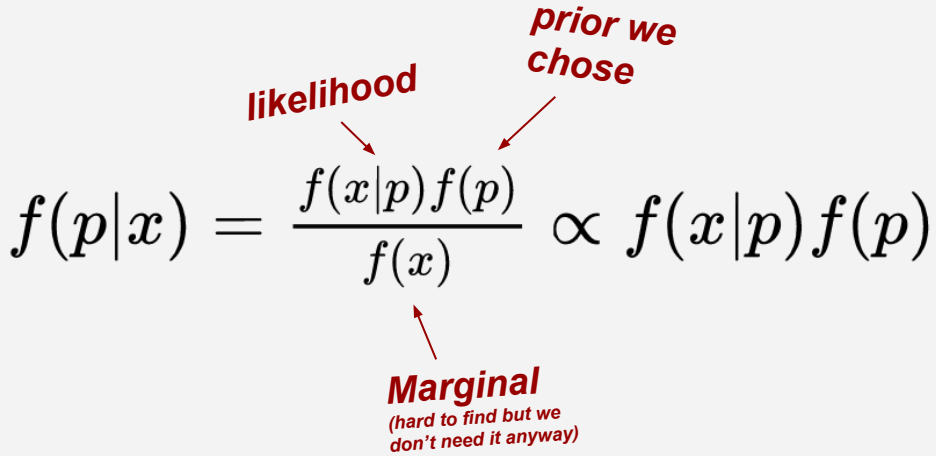
Medium-strength prior belief



Presidential Election

f(*p*) is our **Bayesian prior** - it incorporates our prior beliefs and assumptions into our model. We can make it as strong or as weak of an assumption as we'd like.

Our goal is to find *f*(*p*|*x*), our **posterior distribution**. This is the result of reconciling our priors (preconceived notions) with our data.



Presidential Election

$$
\begin{aligned}
f(p|x) &\propto f(x|p)f(p) \\
&= p^x(1-p)^{1-x} \times cp^{5-1}(1-p)^{5-1} \\
&\propto p^{x+4}(1-p)^{5-x} \\
&= p^{(x+5)-1}(1-p)^{(6-x)-1} \\
&\implies \text{Beta}(x+5, 6-x)
\end{aligned}
$$

Presidential Election

So, the probability that the candidate wins, given what polls say, follows a beta distribution with parameters $x + 5$ and $6 - x$.

The Beta($x + 5$, $6 - x$) distribution has mean $(x + 5) / 11$. So now we can say:

If polls indicate Candidate A wins, we expect their chances are 6/11.

Otherwise, we expect their chances are 5/11.

Well, that was interesting...

Our prior and posterior had the same distribution! This property is known as **conjugacy**. We say that the beta distribution is the **conjugate prior** to the binomial distribution.

There are many such conjugacies. Conjugacy is important, because it allows students to discover posteriors **by hand**. Conjugate pairs allow for a large set of pen-and-paper example problems for students.

Some Conjugate Pairs

Prior	***Likelihood***
Beta	Binomial
Beta	Geometric/Negative Binomial
Exponential/Gamma	Poisson
Normal	Normal (prior on μ)
Gamma	Normal (prior on $1/\sigma^2$)
Gamma	Gamma (prior on α)
Gamma	Gamma (prior on $1/\beta$)

When Conjugacy Fails

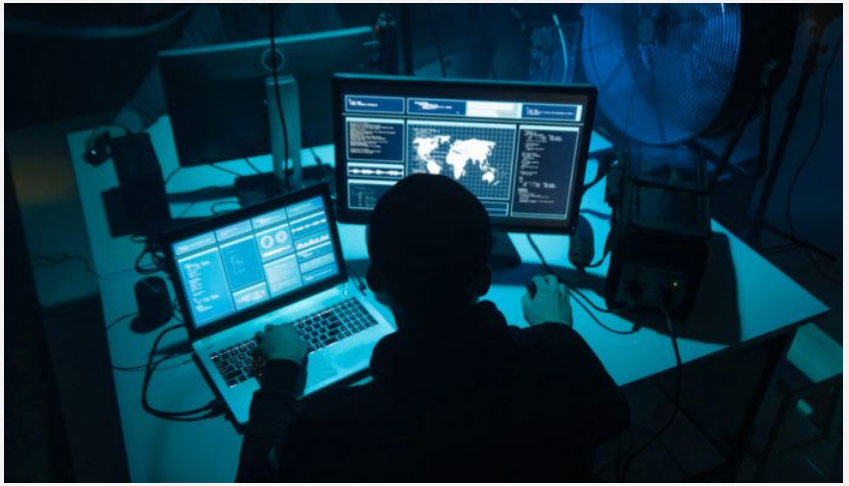
As nice as these examples are, more often than not, **we do not have conjugacy**.

In addition, if a model is more complex than the one-layer hierarchy shown earlier, finding posteriors by hand (if possible) becomes so difficult that it is no longer instructive.

It is time for computers to solve our problems.

Programming Intermezzo

Week 2



Why torture students like this?

Your students will come from a variety of different majors and backgrounds.

Learning to code (if they don't know how already) will be an appreciated break from theory.

Learning to code in R will allow students to bring their theory into application.

Regardless of their background, learning even a little R will make them that much more job-ready. And isn't that what college is all about?

Take it from us...

It’s our job to take people from diverse backgrounds and turn them into functioning statisticians/data scientists in 12 weeks. These are **facts** about the current job market:

- There are no practicing statisticians in 2019 that don’t code.
- College graduates that know **both** statistics **and** programming are in high demand.
- The demand for statisticians that can code is **growing**.

Choice of Language

The de facto beginner programming language for statisticians is **R**. Another viable option is Python.

Our decision against Python was influenced by the fact that it is a general-purpose programming language. It has slightly more overhead that would need to be learned *before* learning statistical programming.

Furthermore, **RStudio** allows students to get up and running with R in a matter of seconds.

Can you really teach programming in a week?

Actually, yes!

However, time is short. R users likely do not need to be full-fledged computer scientists. A surprising amount of fundamental programming can be glossed over or skipped entirely.

Challenge: Teach Programming in One Week

Important:

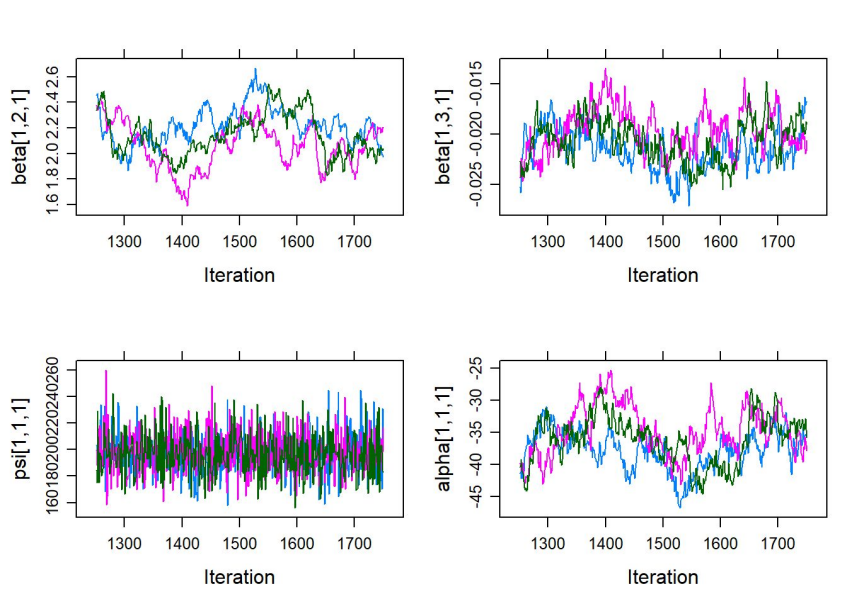
- Native data types
- Dataframe manipulation
- Functions (using)

Not Immediately Important:

- Conditional statements
- Loops
- Functions (making)

MCMC

Week 3


beta[1,2,1]
1.6 1.8 2.0 2.2 2.4 2.6
1300 1400 1500 1600 1700
Iteration
beta[1,3,1]
-0.025 -0.020 -0.015
1300 1400 1500 1600 1700
Iteration
psi[1,1,1]
1601802002202402602
1300 1400 1500 1600 1700
Iteration
alpha[1,1,1]
-45 -40 -35 -30 -25
1300 1400 1500 1600 1700
Iteration

No Conjugacy, No Problem!

In real-world Bayesian problems, we rarely have conjugacy. Without conjugacy, by-hand calculation of posteriors becomes difficult and oftentimes impossible.

Luckily, we can use **Markov chain Monte Carlo (MCMC)** methods to get our computers to do the work for us!

What does MCMC do?

MCMC allows us to sample from our posterior distributions without ever finding them explicitly.

Any estimates we want on our posterior can then be computed empirically on simulated data!

How does it work?

MCMC is a class of algorithms with many variants. The simplest and most common for Bayesian estimation is the **Metropolis-Hastings Algorithm**, which has this general structure:

1. Initialize random starting points
2. Sample from the distribution
3. Calculate an “acceptance ratio” α - a number which describes how confident we are that this new sample is a step in the right direction
4. Accept or reject this new step with probability of our acceptance ratio alpha.
    a. If we accept, this new point is our next point in the process
    b. If we reject, we use the current point again during the next step
5. Repeat steps 2-4 for as long as you can

Let’s start with an Example

Suppose, given a mother’s height, we would like to predict how tall her biological daughter will grow. That is, let *X* be the height of the mother, and let *Y* be the height of the daughter.

We might like to carry out a simple linear regression here:

$$Y = \beta_0 + \beta_1 X + \varepsilon$$

where

$$\varepsilon \sim N(0, \sigma^2)$$

Regression Example

This is more accurately written

$$Y|\beta_0, \beta_1, \sigma = \beta_0 + \beta_1 X + \varepsilon$$

Also, because a constant added to a normal is that same normal distribution shifted, we can write:

$$Y|\beta_0, \beta_1, \sigma \sim \mathcal{N}(\beta_0 + \beta_1 X, \sigma^2)$$

This is a very natural Bayesian situation! We have three parameters on which to put priors.

Our Priors

$$\beta_0 \sim \mathcal{N}(0, \infty)$$ ⟶ ***Uninformative prior - we have no intuition here!***

$$\beta_1 \sim \mathcal{N}(1, 0.1)$$ ⟶ ***Informed prior! We think parents and children will on average have approximately the same height.***

$$\sigma^2 \sim \text{Exponential}\left(\lambda = \frac{1}{3}\right)$$ ⟶ ***Moderately informative prior. We think our expected mean error will be about 3 inches.***

Summary

$$Y|\beta_0, \beta_1, \sigma = \beta_0 + \beta_1 X + \varepsilon$$

$$\beta_0 \sim \mathcal{N}(0, \infty)$$
$$\beta_1 \sim \mathcal{N}(1, 0.1)$$
$$\sigma^2 \sim \text{Exponential}\left(\lambda = \tfrac{1}{3}\right)$$

The situation is simple! The hard part is getting this into software so we can actually compute our **three** posteriors.

Choice of Technology

There is no shortage of MCMC libraries out there. We have decided to use **Stan**.

Why? Right now, Stan is the most popular Bayesian framework. It has APIs in all of the most relevant programming languages (R, Python, MATLAB, Julia, Stata). It even has its own conference, **StanCon**!

Stan has its own syntax that is the same across implementations. Only the language-specific boilerplate differs slightly.

We'll be using R for this workshop, but **we'll have code posted in both R and Python.**

Code

The code for the following example can be found at

https://github.com/timbook/uscots-bayes-workshop

Stan Part 1: The Inputs

model.stan

```
data {
  int<lower=0> J;
  vector[J] x;
  vector[J] y;
}
```

Stan Part 2: The Variables

model.stan

```
parameters {
  real b0;
  real b1;
  real<lower=0> sigma;
}
```

Stan Part 3: The Model

model.stan

```
model {
  // Priors
  b0 ~ normal(0, 100);
  b1 ~ normal(1, 0.1);
  sigma ~ exponential(1/3.0);
  // Likelihood
  y ~ normal(b0 + b1*x, sigma);
}
```

R Implementation: Libraries and Data

bayes-reg.R

```r
library(rstan)
library(alr3)


# Import data - mothers' vs daughters' heights.
data(heights)
```

R Implementation: Libraries and Data

bayes-reg.R

```
head(heights)
```

```
  Mheight Dheight
1    59.7    55.1
2    58.2    56.5
3    60.6    56.0
4    60.7    56.8
5    61.8    56.0
6    55.5    57.9
```

R Implementation: The Model

bayes-reg.R

```
# Model input.
model_data <- list(
  J = nrow(heights),
  x = heights$Mheight,
  y = heights$Dheight
)
```

R Implementation: The Model

bayes-reg.R

```
# Fitting the model!
model <- stan_model("model.stan")
model_fit <- sampling(
    model,
    data = model_data,
    cores = 4
)
```

R Implementation: The Output

Students are now free to analyze various parts of the output and make conclusions on their own! The Stan library provides lots of common Bayesian model metrics out of the box.

bayes-reg.R

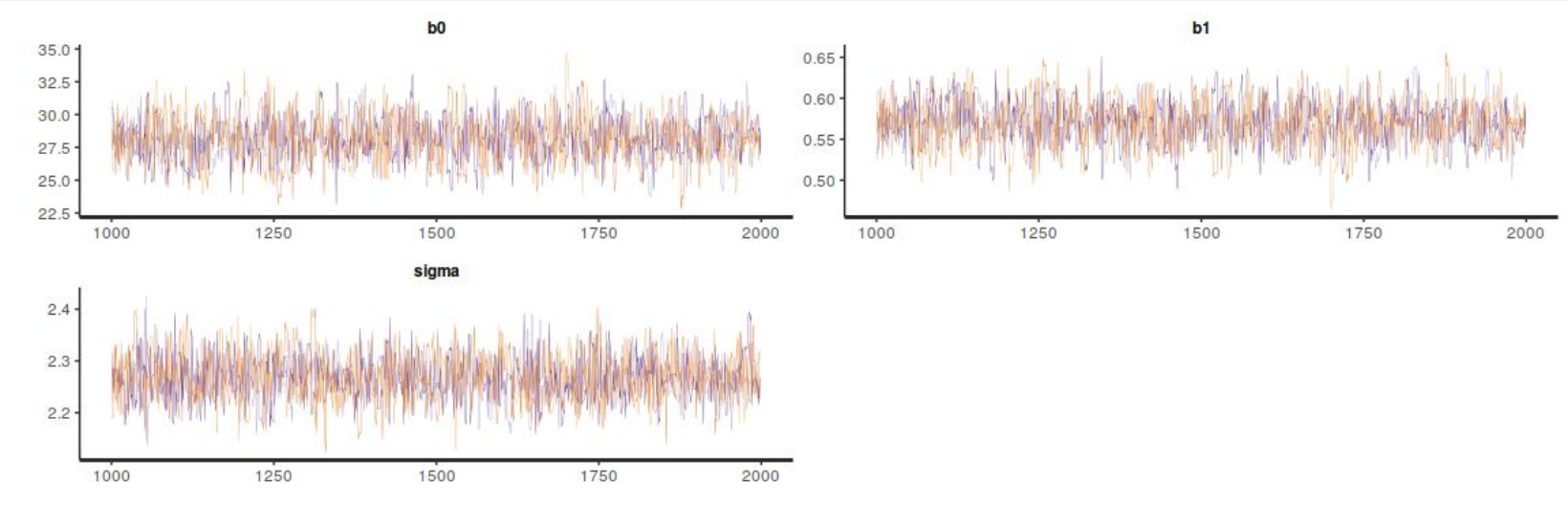
```
summary(model_fit)$summary
```

```
            mean      se_mean          sd          2.5% ...      97.5% ...
b0       28.0838647 0.0419593723 1.56240489 25.0140454 ... 31.0701382 ...
b1        0.5710975 0.0006669647 0.02498515  0.5231919 ...  0.6202006 ...
sigma     2.2707834 0.0010925071 0.04360051  2.1889603 ...  2.3574570 ...
```

R Implementation: The Output

bayes-reg.R

```
traceplot(model_fit)
```



R Implementation: The Output

They can even analyze the simulations and discover their own metrics!

bayes-reg.R

```
as.data.frame(model_fit)
```

```
        b0        b1    sigma      lp__
1 27.93530 0.5730703 2.317833 -1822.056
2 26.75390 0.5931529 2.251934 -1822.039
3 28.30315 0.5682003 2.272517 -1821.483
4 28.33443 0.5672776 2.270647 -1821.309
5 28.89798 0.5576805 2.279194 -1821.541
6 29.12868 0.5545457 2.295129 -1821.735
```

What now?

Now we get to ask the types of questions frequentists can't answer.

- What is the probability the slope is above 0.6?
- What is the probability that a woman who is 5'6" has a daughter who grows to be taller than her?
- What is a 95% credible interval for the slope?
- What is a 95% credible interval for the height of a daughter whose mother is 5'8"?

Note that if you chose not to incorporate programming into your course, all of these questions are answerable if you supply the posterior simulations to the students directly!

Example for you:

According to Brita, your filters should last for 2 months before needing to be replaced (ie, 6 filters/year). However, over the last 6 years, these are the number of filters you've used:

4	**7**	**8**	**6**	**5**	**5**

Is Brita's claim is accurate? (to within ±½ year)

Brita Example

Of course, there are many solutions to this problem. Since we are dealing with count data over a fixed period of time, we assumed the number of filters per yer was Poisson distributed.

We also put an Exponential($\lambda_0 = ⅙$) prior on the Poisson parameter.

Solution code can be found in the Git repository.

$$P(5.5 \leq \lambda \leq 6.5 | \mathbf{x}) = 0.375$$

Eggs Example:

Suppose you work at Corner Room and you're cracking eggs to make omelettes. You notice the current shipment of egg crates is more damaged than normal. You look through 10 egg crates (one dozen eggs each) and note the numbers of cracked eggs in each:

0	1	1	2	0
3	2	3	2	0

What is the probability that the next create of eggs will have no cracks?

Eggs Example:

With the following parametrization:

$$X|p \sim \text{Bin}(12, p)$$

$$p \sim \text{Beta}(2, 10)$$

Corresponds to ⅙ mean

We had a posterior mean ***p = 0.12***.

$$P(X = 0|p = 0.12) = 0.214$$

Conclusions

- Bayesian statistics already fits neatly into a pre-existing mathematical statistics curriculum.
- Teaching programming fundamentals to undergraduates is pretty important on its own - but skippable.
- The Bayesian toolbox allows us to ask more human-understandable questions with intuitive answer.
- Teaching only frequentism tells only half the story! Bayesian statistics offer a flexibility found nowhere else in the statistical world.

Thank you!

Matthew Brems

General Assembly
matthew.w.brems@gmail.com

Timothy Book

General Assembly
timothykbook@gmail.com