# An introduction to Bayesian regression

Guest lecture for Psych 610/710 at UW–Madison

---

Tristan Mahr

April 13, 2017

# Online resources

Slides and R code that produced them are online:
`https://github.com/tjmahr/Psych710_BayesLecture`

I gave a similar, more code-heavy version of this talk to the R Users Group: `https://github.com/tjmahr/MadR_RStanARM`

# Overview

- A little about me and how I got into Bayes
- Mathematical intuition building
- Bayesian updating
- Fitting a model with RStanARM
- Big takeaway ideas

# Background

# About me

- I am dissertator in Communication Sciences and Disorders
- I study word recognition in preschoolers
- For statistics, I mostly do multilevel logistic regression models
- R enthusiast

# I was once in your shoes

I learned stats and R in this course with Markus Brauer and John Curtin.

I *still* refer to the slides from this course on contrast codes.

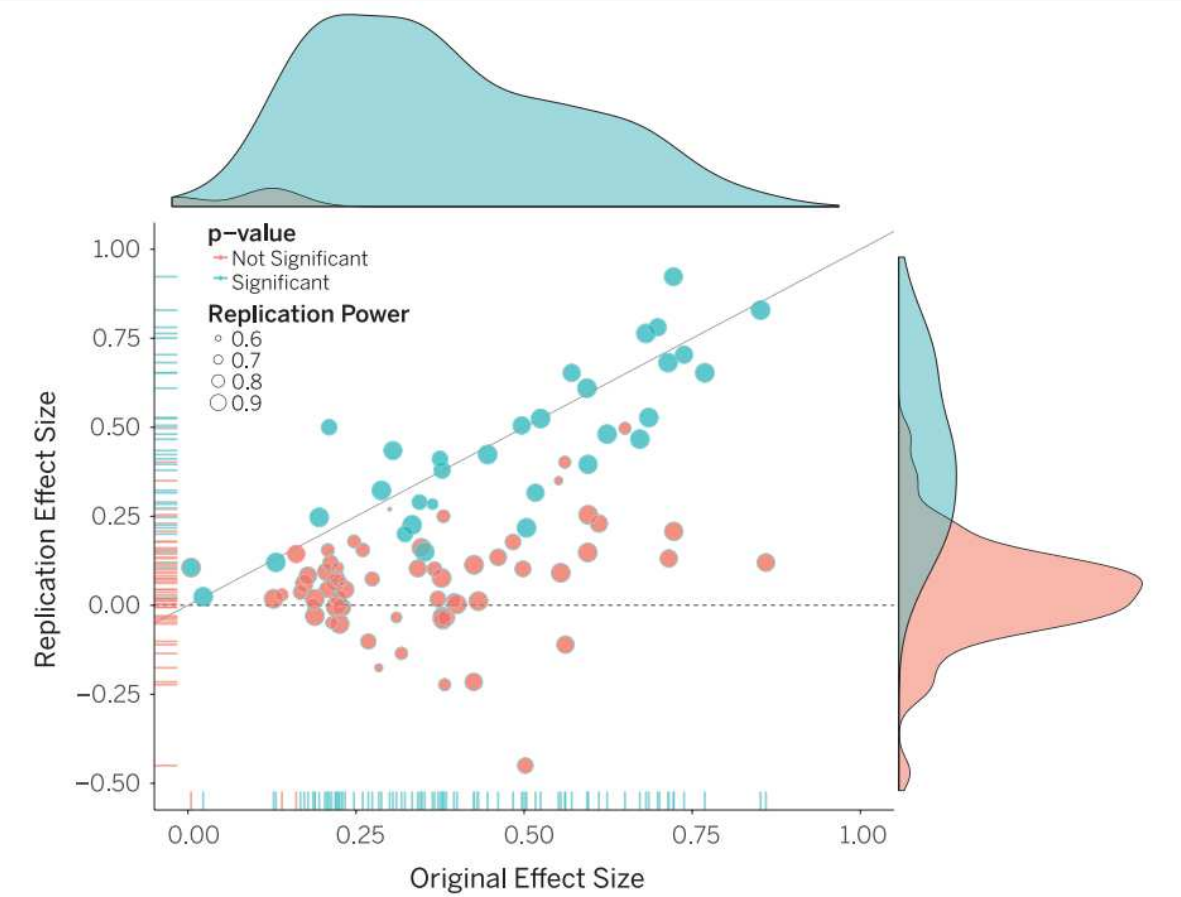
But now I'm a "Bayesian".

# A timeline

# August 2015: The "Crisis" in Psychology

Open Science Collaboration (2015) tries to replicate 100 studies published in 3 psychology different journals in 2008.

- Boil a study down to 1 test statistic and 1 effect size.
- Replicate the study.
- Compare replication's test statistic and effect size against original.




**Original study effect size versus replication effect size (correlation coefficients).** Diagonal line represents replication effect size equal to original effect size. Dotted line represents replication effect size of 0. Points below the dotted line were effects in the opposite direction of the original. Density plots are separated by significant (blue) and nonsignificant (red) effects.

**Figure 1:** Scatter plot of original vs replicated effect sizes

- Approximately 36% of the studies are replicated (same test statistic).
- On average, effect sizes in replications are half that of the original studies.

# Reactions

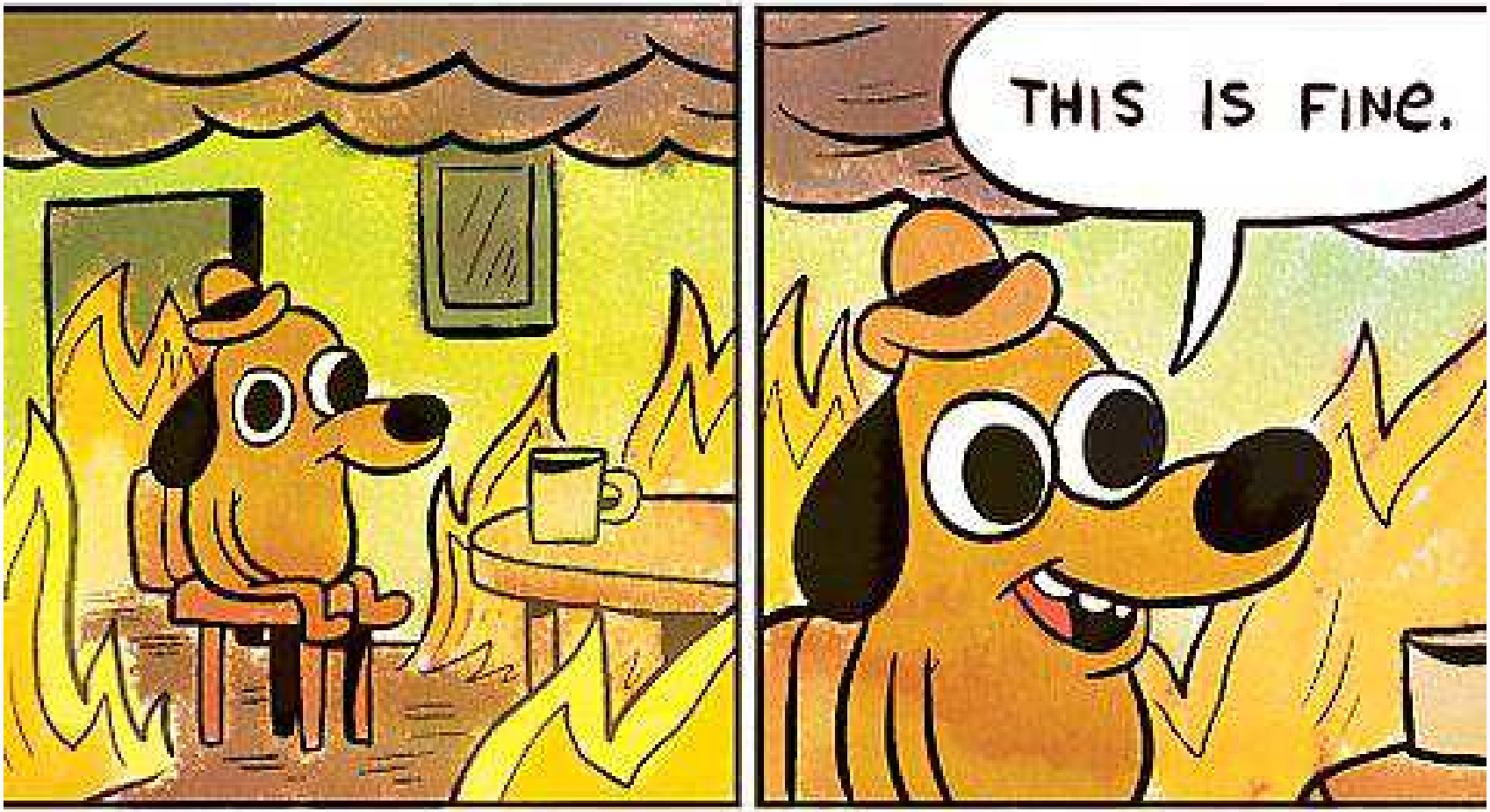



**Figure 2:** I don't know how to turn off the figure labeling feature

# Reactions

- We're doomed.

# Reactions

- We're doomed.
- Most findings are probably false, and we knew that already.

# Reactions

- We're doomed.
- Most findings are probably false, and we knew that already.
- No, this is business as usual.

# Reactions

- We're doomed.
- Most findings are probably false, and we knew that already.
- No, this is business as usual.
- Any credible discipline has to do this kind of house-cleaning from time to time.

# Lots of hand wringing and soul searching

Some reactionary:

- Replication creates an industry for incompetent hacks.
- Here come the methodological terrorists!

Some constructive:

- Everything is f'ed – so what else is new?
- Increased rigor and openness are a good thing.

# Crisis made me think more about questionable practices

All those unintentional acts and rituals to appease the Statistical Significance gods.

# Crisis made me think more about questionable practices

All those unintentional acts and rituals to appease the Statistical Significance gods.

**HARKing** Hypothesizing after results are known.
Telling a story to fit the data.

# Crisis made me think more about questionable practices

All those unintentional acts and rituals to appease the Statistical Significance gods.

**HARKing** Hypothesizing after results are known.
Telling a story to fit the data.

**Garden of forking data** Conducting countless sub-tests and sub-analyses on the data.

# Crisis made me think more about questionable practices

All those unintentional acts and rituals to appease the Statistical Significance gods.

**HARKing** Hypothesizing after results are known. Telling a story to fit the data.

**Garden of forking data** Conducting countless sub-tests and sub-analyses on the data.

***p*-hacking** Doing these tests in order to find a significant effect.

# Crisis made me think more about questionable practices

All those unintentional acts and rituals to appease the Statistical Significance gods.

**HARKing** Hypothesizing after results are known. Telling a story to fit the data.

**Garden of forking data** Conducting countless sub-tests and sub-analyses on the data.

***p*-hacking** Doing these tests in order to find a significant effect.

**Selective reporting** Reporting only the tests that yielded a significant result.

# My sense

The usual way of doing things is *insecure*.

# My sense

The usual way of doing things is *insecure*.

- Perfectly fine if you know what you're doing.

# My sense

The usual way of doing things is *insecure*.

- Perfectly fine if you know what you're doing.
- Works great if you pre-register analyses. Provides error control.

# My sense

The usual way of doing things is *insecure*.

- Perfectly fine if you know what you're doing.
- Works great if you pre-register analyses. Provides error control.
- But vulnerable to exploitation.

# My sense

The usual way of doing things is *insecure*.

- Perfectly fine if you know what you're doing.
- Works great if you pre-register analyses. Provides error control.
- But vulnerable to exploitation.
- And many people don't know what they're doing.

# My response to the crisis

I want to avoid these questionable practices.

I want to level up my stats and explore new techniques.

- Maybe more robust estimation techniques?
- Maybe machine learning techniques to complement conventional analyses?

# My response to the crisis

I want to avoid these questionable practices.

I want to level up my stats and explore new techniques.

- Maybe more robust estimation techniques?
- Maybe machine learning techniques to complement conventional analyses?

I want something less finicky than statistical significance.

- *p*-values don't mean what many people think they mean.
- Neither do confidence intervals.
- Statistical significance is not related to practical significance.

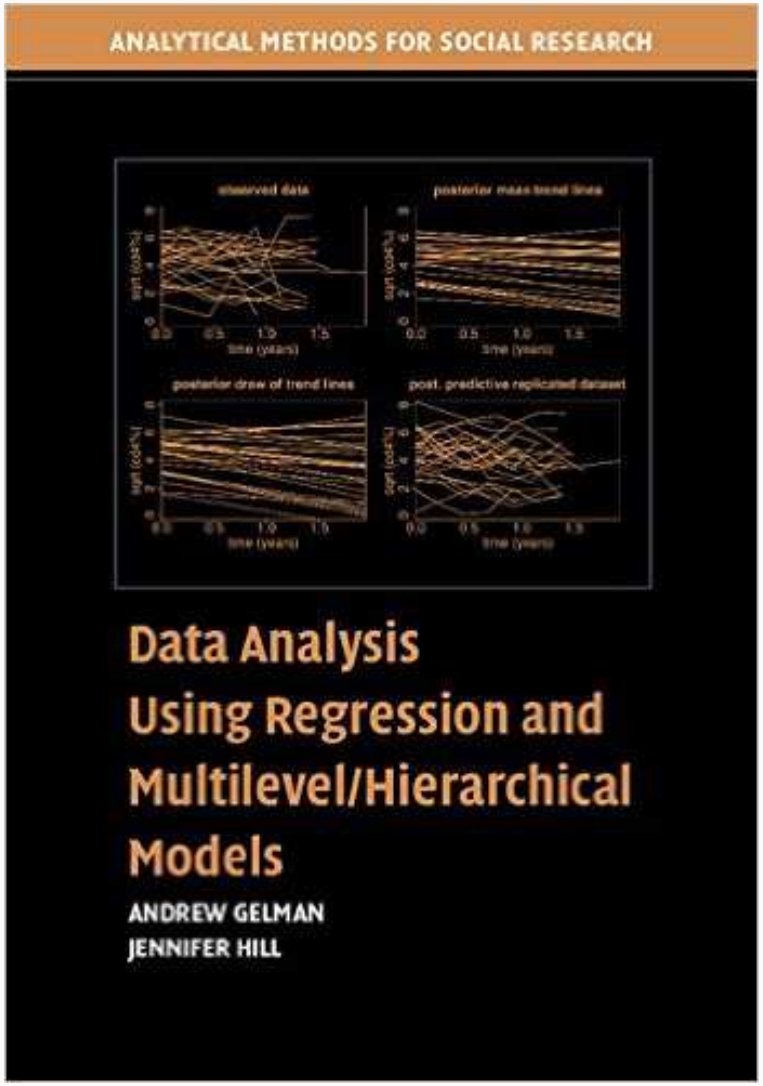



**Figure 3:** Cover of Data Analysis USing Regression and Multilevel/Hierarchical Models

I started reading the Gelman and Hill book.

- This is the book for the `arm` package.
- Still the best treatment of multilevel models in R despite being 10 years old.

It emphasizes estimation, uncertainty and simulation.

I started reading the Gelman and Hill book.

- This is the book for the `arm` package.
- Still the best treatment of multilevel models in R despite being 10 years old.

It emphasizes estimation, uncertainty and simulation.

Midway through, the book pivots to Bayesian estimation.
(Multilevel models are kinda Bayesian because they borrow information across different clusters.)

# January 2016

I'm down a rabbit hole, writing Stan (Bayesian) models to fit the models from the ARM book, and there is an influx of Bayesian tools for R.

- Statistical Rethinking, a book that reteaches regression from a Bayesian perspective with R and Stan, is released.
- New version of brms is released. This package converts R model code into Stan programs.
- RStanARM is released.
- A blog post circulates: "R Users Will Now Inevitably Become Bayesians".

I eat all this up. I become a convert.

# Long story short

The replication crisis sparked my curiosity, and a wave of new tools and resources made it really easy to get started with Bayesian stats.

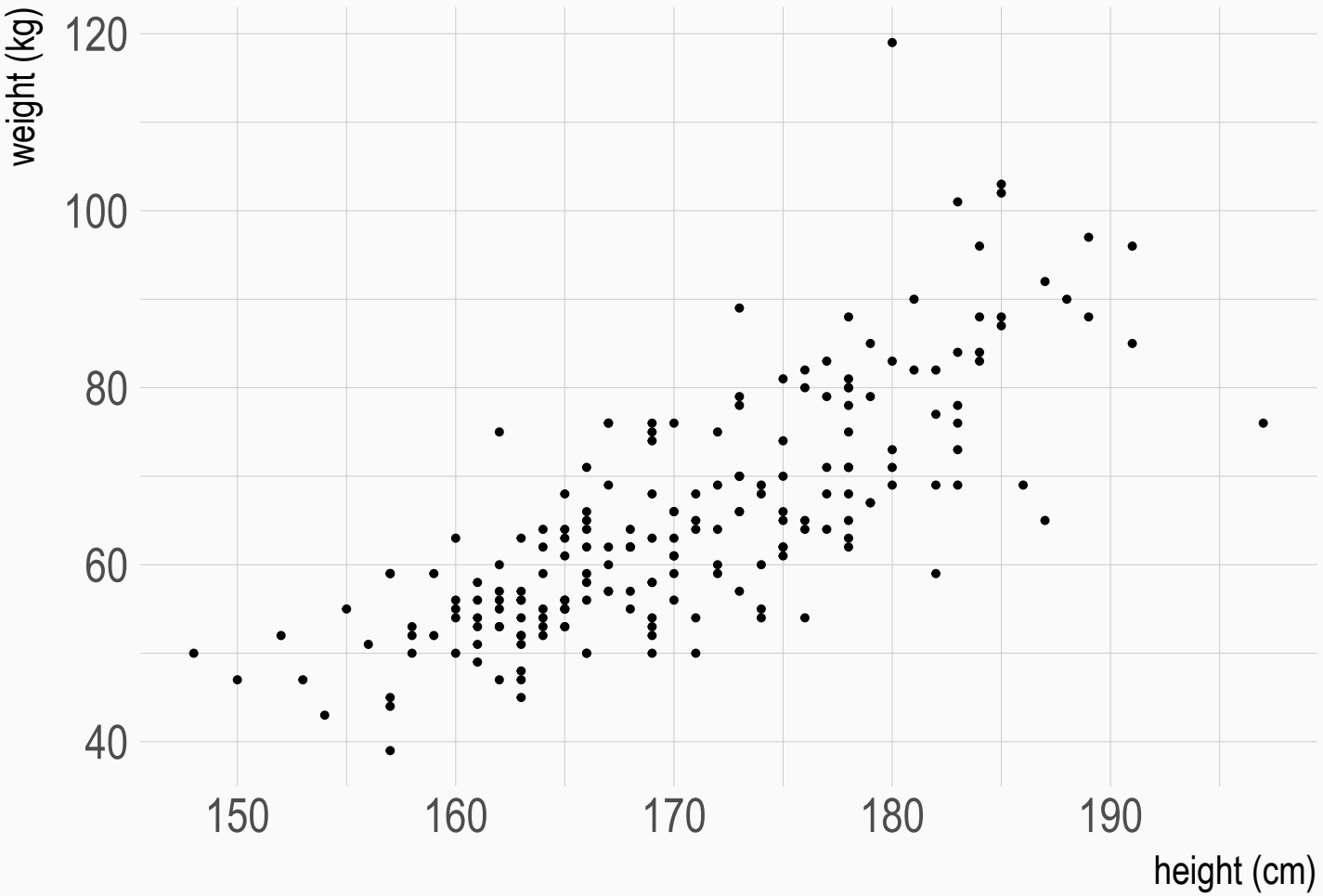
My goal with this approach has been to make better, more honest scientific summaries of observed data.

# Classical regression versus Bayesian regression in a few plots

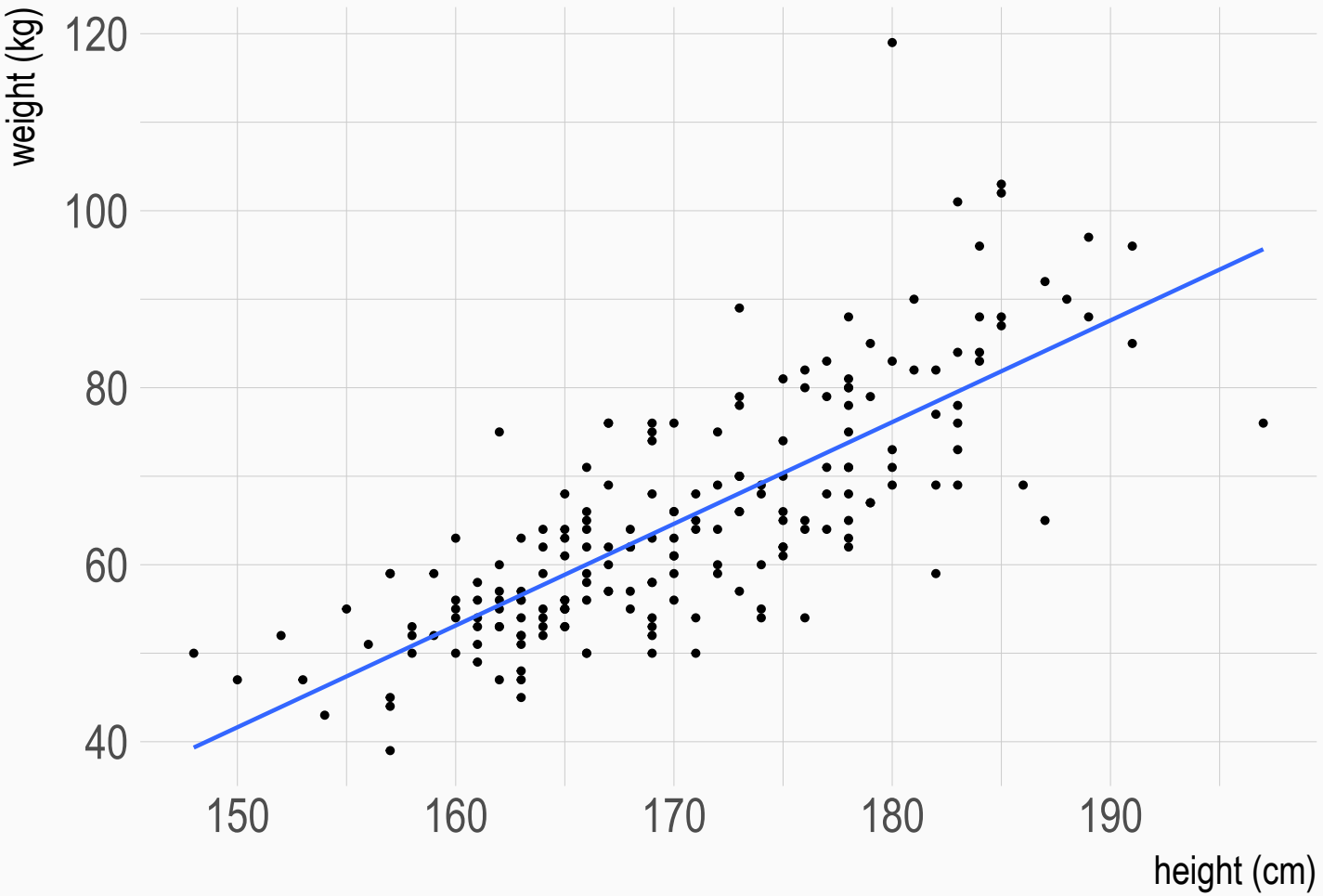
# The data

```
# Some toy data
davis <- car::Davis %>% filter(100 < height) %>% as_data_frame()
davis
#> # A tibble: 199 × 5
#>       sex weight height repwt repht
#>    <fctr>  <int>  <int> <int> <int>
#> 1       M     77    182    77   180
#> 2       F     58    161    51   159
#> 3       F     53    161    54   158
#> 4       M     68    177    70   175
#> 5       F     59    157    59   155
#> 6       M     76    170    76   165
#> 7       M     76    167    77   165
#> 8       M     69    186    73   180
#> 9       M     71    178    71   175
#> 10      M     65    171    64   170
#> # ... with 189 more rows
```

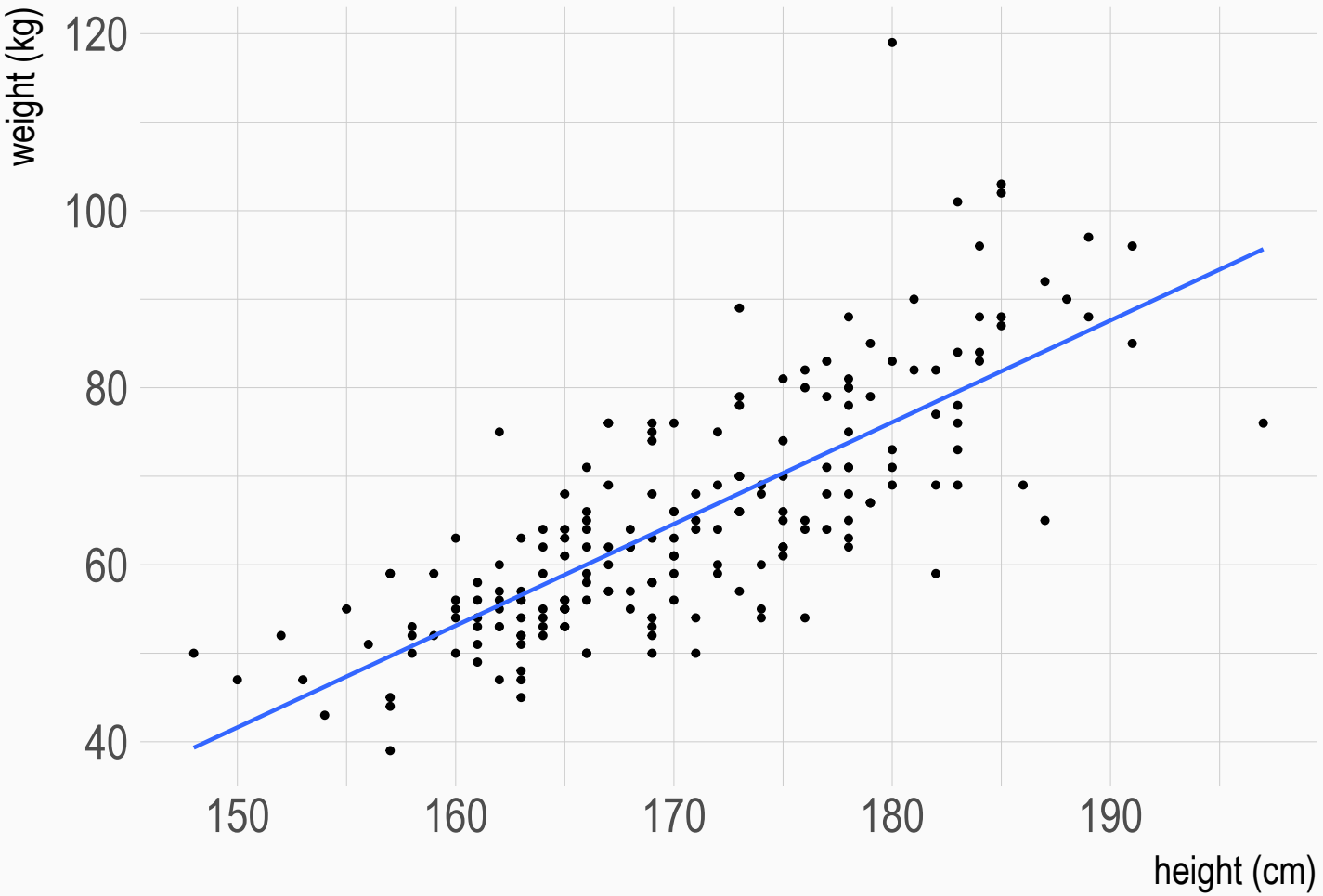
# The data



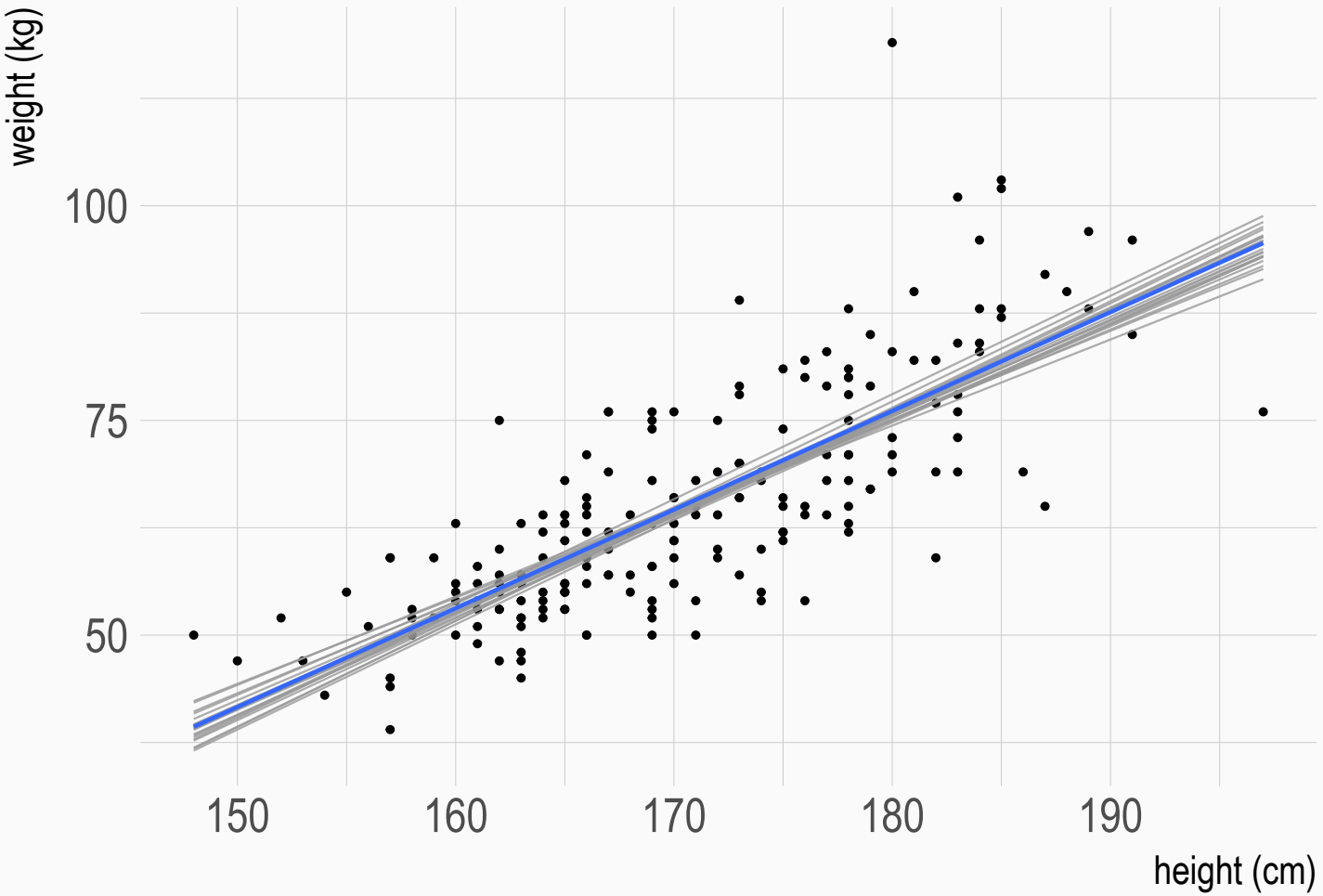
# Classical model provides the line of best fit



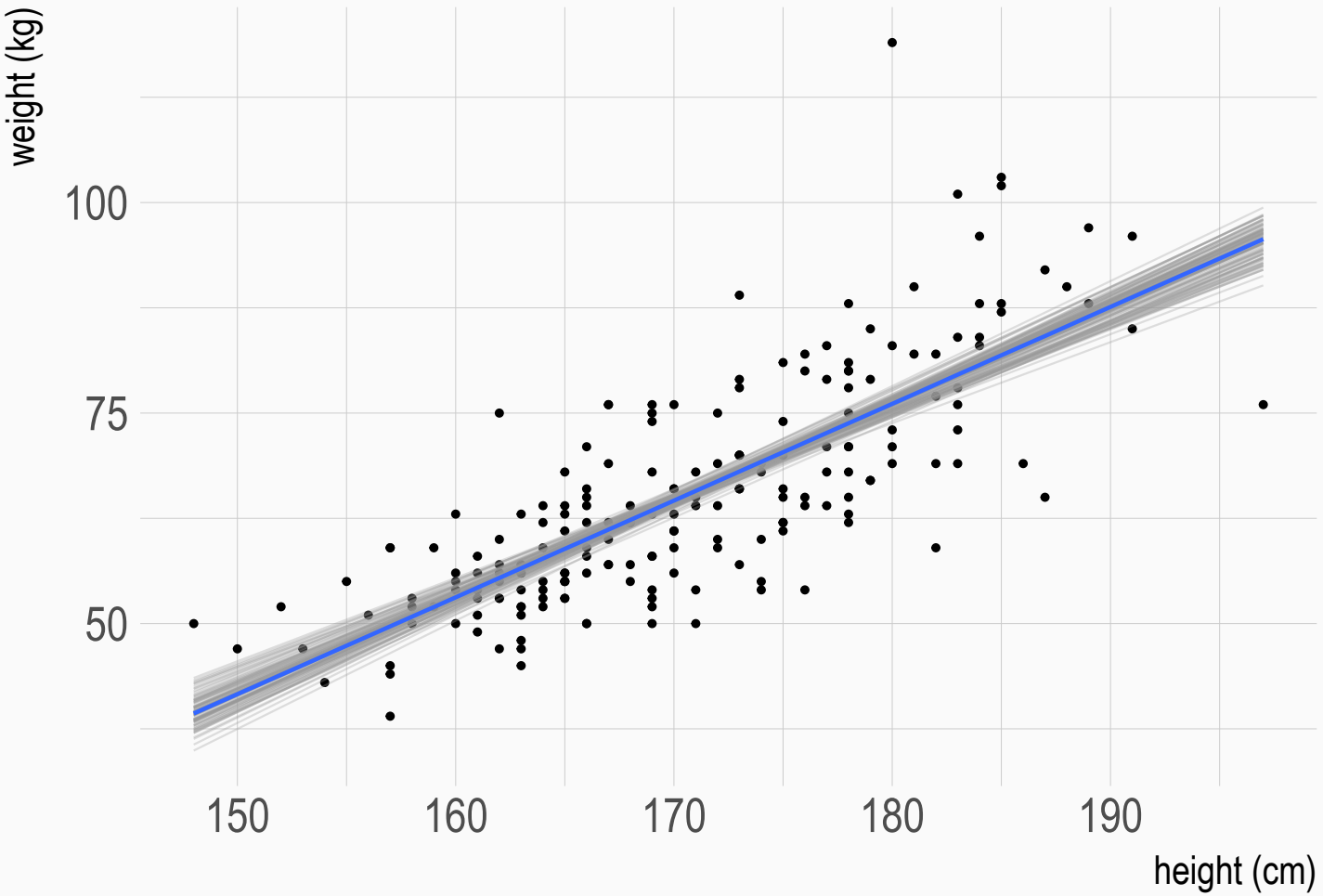
# Bayesian model's median line of fit



# Median line and 20 other lines from posterior



# Median line and 100 other lines from posterior



# Summary

- Classical: There is a single "true" line of best fit, and I'll give my best estimate of it.
- Bayesian: There is a distribution of lines of fit—some more plausible than others—and I'll give you samples from that distribution.

# Building mathematical intuitions

# Caveat

- These slides and these examples are meant to illustrate the pieces of Bayes theorem.
- This is not a rigorous mathematical description of Bayesian probability or regression.

# Conditional probability review

$p(A \mid B)$ : probability of A given B

# Conditional probability review

$p(A \mid B)$ : probability of A given B

Suppose that 95% of emails with the phrase "investment opportunity" are spam.

# Conditional probability review

$$p(A \mid B) : \text{probability of A given B}$$

Suppose that 95% of emails with the phrase “investment opportunity” are spam.

$$p(\text{spam email} \mid \text{"investment opportunity"}) = .95$$

# Conditional probability review

What would this probability express?

$$p(\text{"investment opportunity"} \mid \text{spam email})$$

# Conditional probability review

What would this probability express?

$$p(\text{"investment opportunity"} \mid \text{spam email})$$

That ordering matters. $p(A \mid B)$ is not the same as $p(B \mid A)$.

# Bayes' theorem

A theorem about conditional probability.

$$p(B \mid A) = \frac{p(A \mid B) * p(B)}{p(A)}$$

I can never remember this equation with letters. Here's how I prefer to write it.

$$p(\text{hypothesis} \mid \text{data}) = \frac{p(\text{data} \mid \text{hypothesis}) * p(\text{hypothesis})}{p(\text{data})}$$

$$p(\text{hypothesis} \mid \text{data}) = \frac{p(\text{data} \mid \text{hypothesis}) * p(\text{hypothesis})}{p(\text{data})}$$

The "hypothesis" is typically something unobserved or unknown. It's what you want to learn about using the data.

$$p(\text{hypothesis} \mid \text{data}) = \frac{p(\text{data} \mid \text{hypothesis}) * p(\text{hypothesis})}{p(\text{data})}$$

The "hypothesis" is typically something unobserved or unknown. It's what you want to learn about using the data.

For regression models, the "hypothesis" is a parameter (intercept, slopes or error terms).

Bayes theorem tells you the probability of the hypothesis given the data.

# General structure

How plausible is some hypothesis given the data?

$$p(\text{hypothesis} \mid \text{data}) = \frac{p(\text{data} \mid \text{hypothesis}) * p(\text{hypothesis})}{p(\text{data})}$$

# General structure

How plausible is some hypothesis given the data?

$$p(\text{hypothesis} \mid \text{data}) = \frac{p(\text{data} \mid \text{hypothesis}) * p(\text{hypothesis})}{p(\text{data})}$$

Pieces of the equation:

$$\text{posterior} = \frac{\text{likelihood} * \text{prior}}{\text{average likelihood}}$$

# Classifying emails

I got an email with the word “cialis” in it. Is it spam?

# Classifying emails

I got an email with the word "cialis" in it. Is it spam?

- What I want to know is spam-ness (versus ham-ness).
- What I have is an email with the word "cialis".

# Classifying emails

I got an email with the word “cialis” in it. Is it spam?

- What I want to know is spam-ness (versus ham-ness).
- What I have is an email with the word “cialis”.

$$P(\text{spam} \mid \text{"cialis"}) = \frac{P(\text{"cialis"} \mid \text{spam}) * P(\text{spam})}{P(\text{"cialis"})}$$

# Email example

The two unconditional probabilities are base rates that need to be accounted for.

# Email example

The two unconditional probabilities are base rates that need to be accounted for.

The prior is the frequency of spam in general. The average likelihood is the frequency of the word “cialis” in emails.

$$P(\text{spam} \mid \text{"cialis"}) = \frac{\text{"cialis" freq. in spam} * \text{spam rate}}{\text{"cialis" freq.}}$$

# “Bayesianism”

Some people would argue that using Bayes theorem is not “Bayesian”. After all, in this example, we’re just counting the frequency of events.

It’s kind of weird, but it is also true.

Simple event-counting is not what people usually mean by the word “Bayesian”.

# The "Bayesianism" form of Bayes' theorem

$$\text{updated information} = \frac{\text{likelihood of data} * \text{prior information}}{\text{average likelihood of data}}$$

Bayes' theorem provides a systematic way to update our knowledge as we encounter new data.

# The "Bayesianism" form of Bayes' theorem

$$\text{updated information} = \frac{\text{likelihood of data} * \text{prior information}}{\text{average likelihood of data}}$$

Bayes' theorem provides a systematic way to update our knowledge as we encounter new data.

$$\text{updated beliefs} \propto \text{likelihood of data} * \text{prior beliefs}$$

- Update your beliefs in proportion to how well the data fits those beliefs.
- Your beliefs have probabilities. You can quantify your uncertainty about what you know.

# Okay, but what is likelihood?

*Sidenote: This is nifty. A lot of my stats training made more sense once I had a broader understanding of likelihood.*

# First, what are models?!

What is a statistical model?

# First, what are models?!

What is a statistical model?

It's a description of how the data could have been generated.

# IQ example

*IQ scores are normally distributed.*

# IQ example

*IQ scores are normally distributed.*

$$\text{IQ}_i \sim \text{Normal}(\underbrace{\mu}_{\text{mean}}, \underbrace{\sigma}_{\text{SD}})$$

(The $\sim$ means "sampled from" or "drawn from".)

$\mu$ and $\sigma$ are parameters for this model that change the center and spread of the normal bell curve.

The normative IQ model has $\mu = 100$ and $\sigma = 15$.

# Likelihood measures fit

How likely are the data in a given model?

# Likelihood measures fit

How likely are the data in a given model?

I never see it explained this way, but I think of likelihood as "fit".

How the well data fits in a given model.

# An IQ example

We found some IQ scores in an old, questionable dataset.

```
library(dplyr)
iqs <- car::Burt$IQbio
iqs
#>  [1]  82  80  88 108 116 117 132  71  75  93  95  88 111  63
#> [15]  77  86  83  93  97  87  94  96 112 113 106 107  98
```

# An IQ example

We found some IQ scores in an old, questionable dataset.

```
library(dplyr)
iqs <- car::Burt$IQbio
iqs
#>  [1]  82  80  88 108 116 117 132  71  75  93  95  88 111  63
#> [15]  77  86  83  93  97  87  94  96 112 113 106 107  98
```

IQs are designed to have a normal distribution with a population mean of 100 and an SD of 15.

How well do these data *fit* in that kind of bell curve?

# Density as height on a bell curve

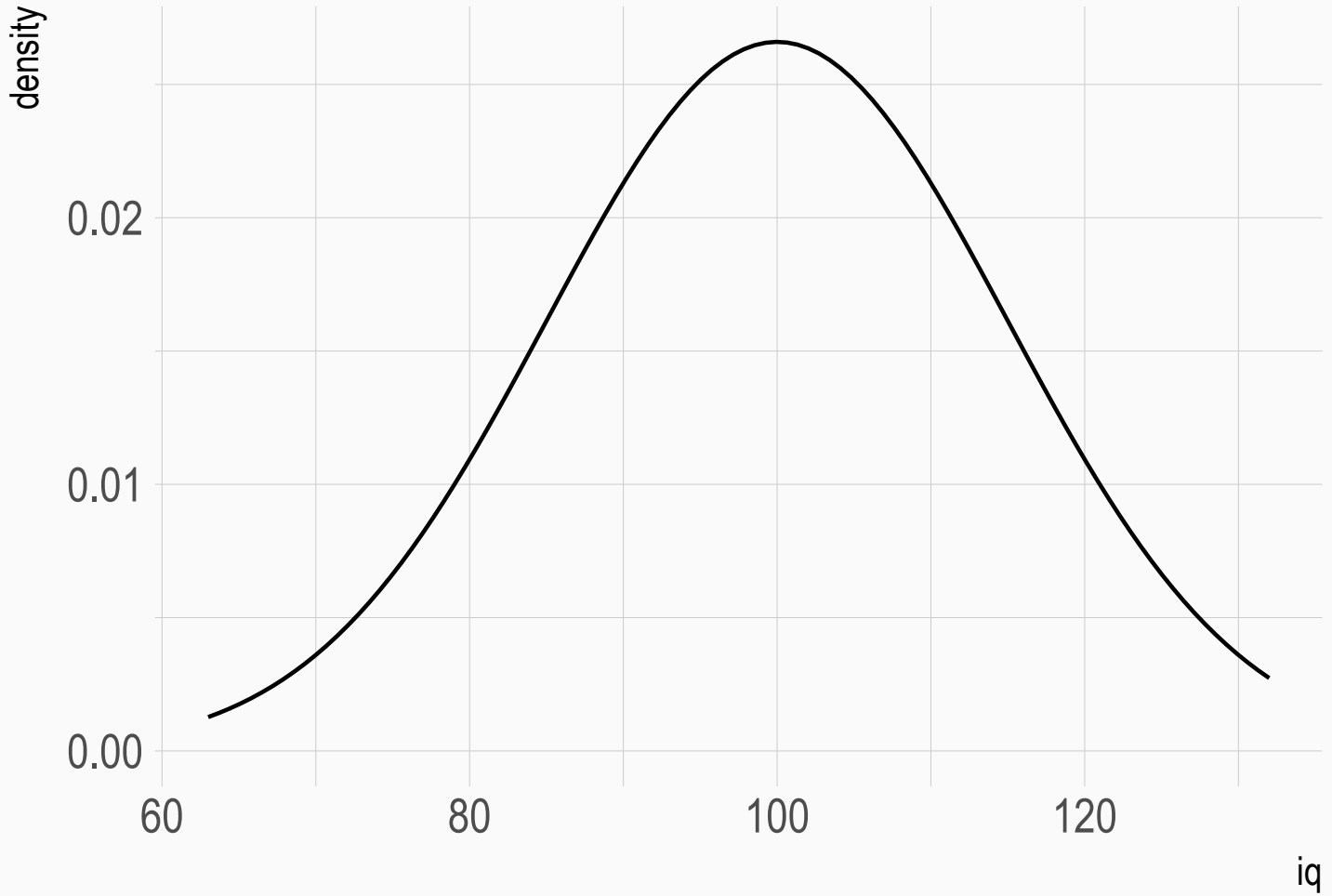



**Figure 4:** A hypothetical bell curve with a mean of 100 and SD of 15.

# Density measures likelihood

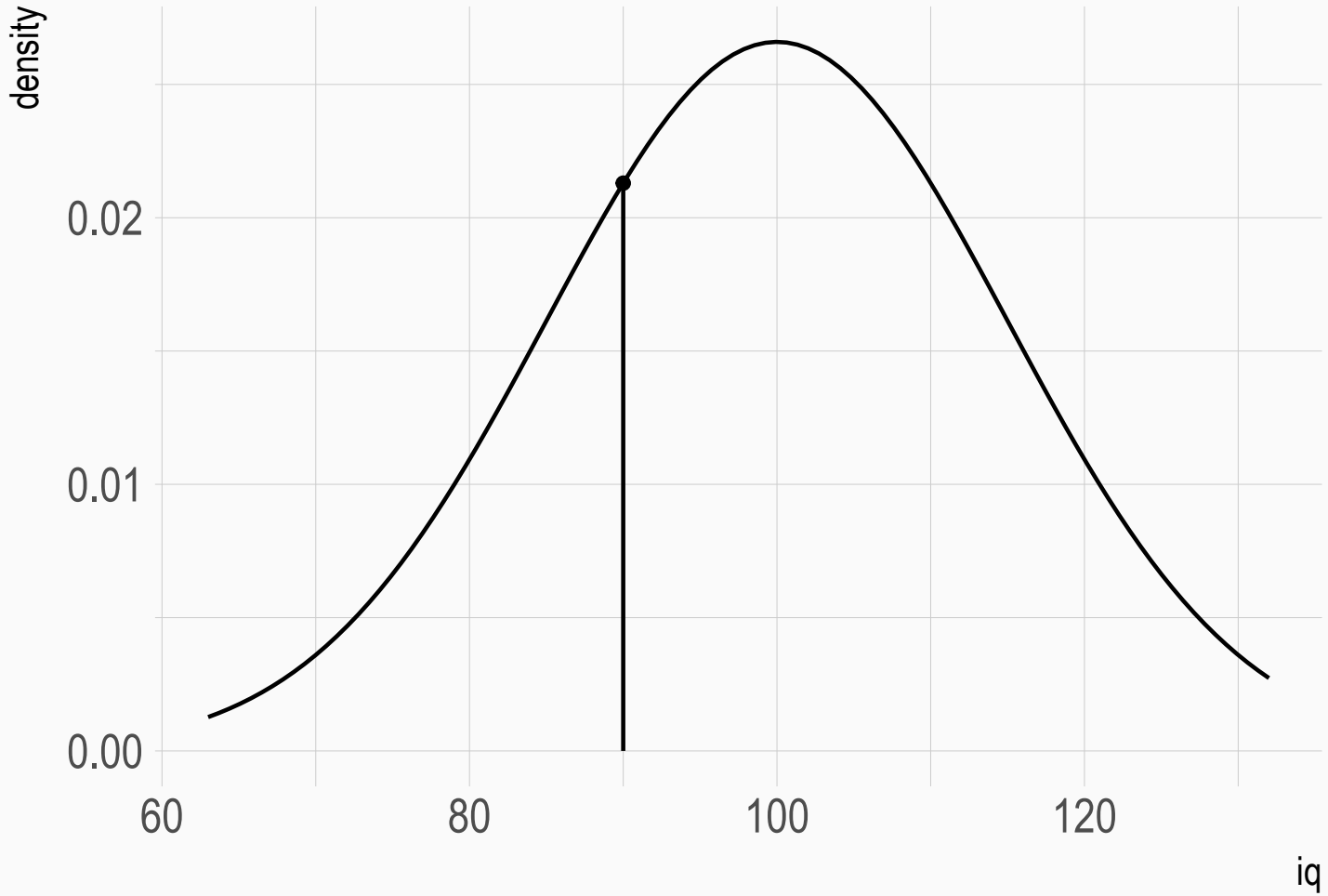



**Figure 5:** Likelihood of an IQ of 90

- Height of each point on curve is density around that point.
- Higher density regions are more likely.
- Data farther from peak density is less likely.

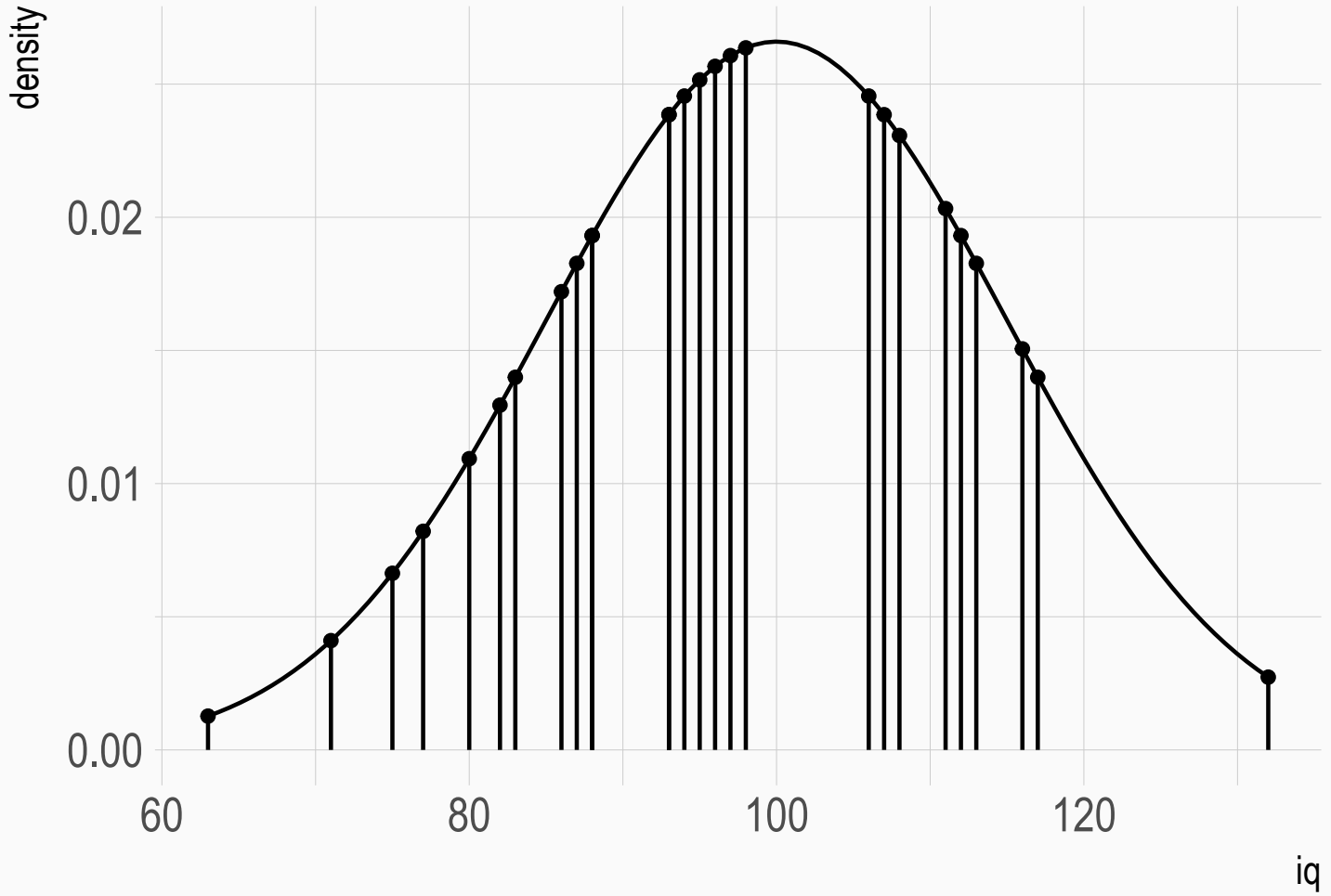



**Figure 6:** Density of IQ scores drawn a bell curve with mean 100.

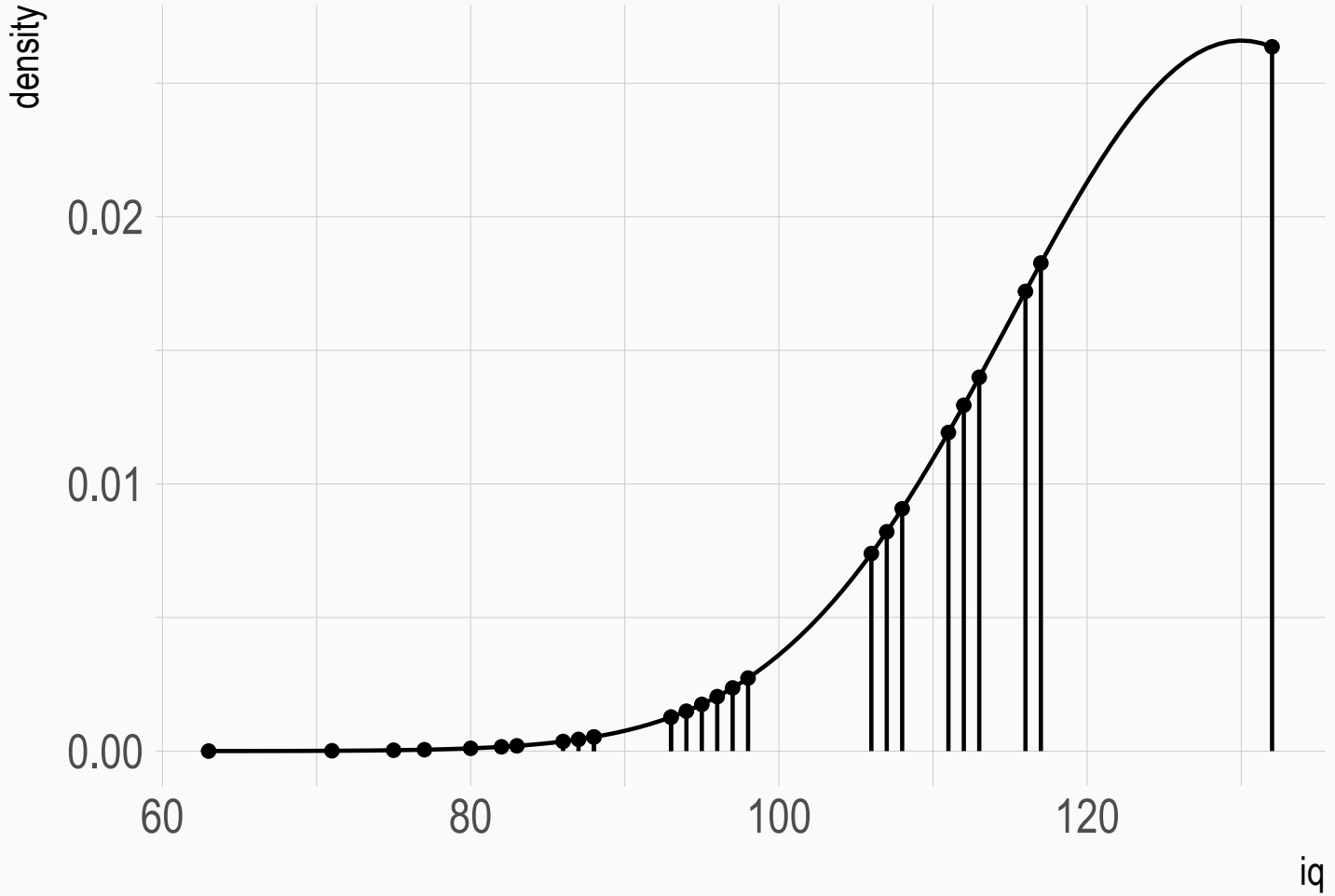



**Figure 7:** Density of IQ scores drawn a bell curve with mean 130. The fit is terrible.

Density function `dnorm(xs, mean = 100, sd = 15)` tells us the height of each value in `xs` when drawn on a normal bell curve.

```
# likelihood (density) of each point
dnorm(iqs, 100, 15) %>% round(3)
#>  [1] 0.013 0.011 0.019 0.023 0.015 0.014 0.003 0.004 0.007
#> [10] 0.024 0.025 0.019 0.020 0.001 0.008 0.017 0.014 0.024
#> [19] 0.026 0.018 0.025 0.026 0.019 0.018 0.025 0.024 0.026
```

Density function `dnorm(xs, mean = 100, sd = 15)` tells us the height of each value in `xs` when drawn on a normal bell curve.

```
# likelihood (density) of each point
dnorm(iqs, 100, 15) %>% round(3)
#>  [1] 0.013 0.011 0.019 0.023 0.015 0.014 0.003 0.004 0.007
#> [10] 0.024 0.025 0.019 0.020 0.001 0.008 0.017 0.014 0.024
#> [19] 0.026 0.018 0.025 0.026 0.019 0.018 0.025 0.024 0.026
```

Likelihood of all points is the product. These quantities get vanishingly small so we sum their logs instead. (Hence, **log-likelihoods**.)

```
# 2 * 10^-50 is vaaaaaaanishingly small!
prod(dnorm(iqs, 100, 15))
#> [1] 2.276823e-50

# log scale
sum(dnorm(iqs, 100, 15, log = TRUE))
#> [1] -114.3065
```

Log-likelihoods provide a measure of how well the data fit a given normal distribution.

Which mean best fits the data? Below average IQ (85), average IQ (100), or above average IQ (115)? (Higher is better.)

Log-likelihoods provide a measure of how well the data fit a given normal distribution.

Which mean best fits the data? Below average IQ (85), average IQ (100), or above average IQ (115)? (Higher is better.)

```
sum(dnorm(iqs, 85, 15, log = TRUE))
#> [1] -119.0065
sum(dnorm(iqs, 100, 15, log = TRUE))
#> [1] -114.3065
sum(dnorm(iqs, 115, 15, log = TRUE))
#> [1] -136.6065
```

```r
sum(dnorm(iqs, 85, 15, log = TRUE))
#> [1] -119.0065
sum(dnorm(iqs, 100, 15, log = TRUE))
#> [1] -114.3065
sum(dnorm(iqs, 115, 15, log = TRUE))
#> [1] -136.6065
```

Of these three, the data fit best with the "population average" mean (100).

We just used a **maximum likelihood** criterion to choose among these alternatives!

# Likelihood summary

We have some model of how the data could be generated. This model has tuneable parameters.

> *The IQs are drawn from a normal distribution with an SD of 15 and some unknown mean.*

Likelihood is how well the observed data fit in a particular data-generating model.

Classical regression's "line of best fit" finds model parameters that maximize the likelihood of the data.

# Bayesian models

# Bayesian updating

Let's consider all integer values from 70 to 130 as equally probable means for the IQs. This is a flat or uniform prior.

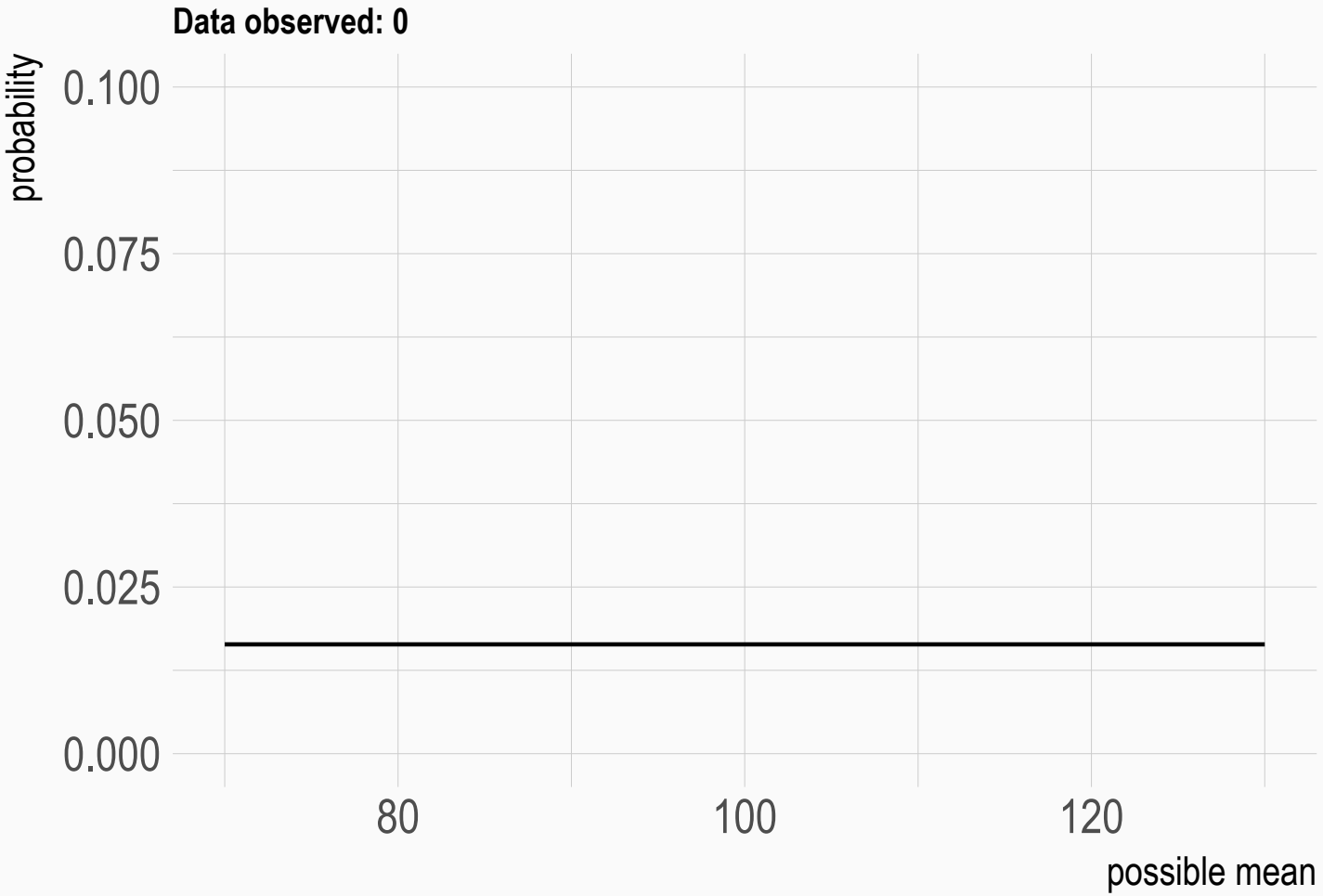
Here's our model.

$$\begin{aligned} \mathrm{IQ}_i &\sim \text{Normal}(\mu, \sigma = 15) && \text{[likelihood]} \\ \mu &\sim \{\text{integers from 70 to 130}\} && \text{[prior for } \mu\text{]} \end{aligned}$$

We are going to use **grid approximation** for this example. That means systematically exploring about a bunch of parameter values. (It's mostly useful for illustrating how Bayes' theorem works.)

```
df_iq_model <- data_frame(
  # Candidate mean value
  mean = 70:130,
  # Probability of each candidate mean right now
  prob = 1 / length(mean),
  # Probability of each candidate mean during the last update
  previous = NA_real_)
```

```r
# Probabilities sum to 1
sum(df_iq_model$prob)
#> [1] 1
df_iq_model
#> # A tibble: 61 × 3
#>     mean       prob previous
#>    <int>      <dbl>    <dbl>
#> 1     70 0.01639344       NA
#> 2     71 0.01639344       NA
#> 3     72 0.01639344       NA
#> 4     73 0.01639344       NA
#> 5     74 0.01639344       NA
#> 6     75 0.01639344       NA
#> 7     76 0.01639344       NA
#> 8     77 0.01639344       NA
#> 9     78 0.01639344       NA
#> 10    79 0.01639344       NA
#> # ... with 51 more rows
```


Data observed: 0
probability
0.100
0.075
0.050
0.025
0.000
80
100
120
possible mean

We observe one data-point, $y = 82$, and update our prior information using the likelihood of the data at each possible mean.

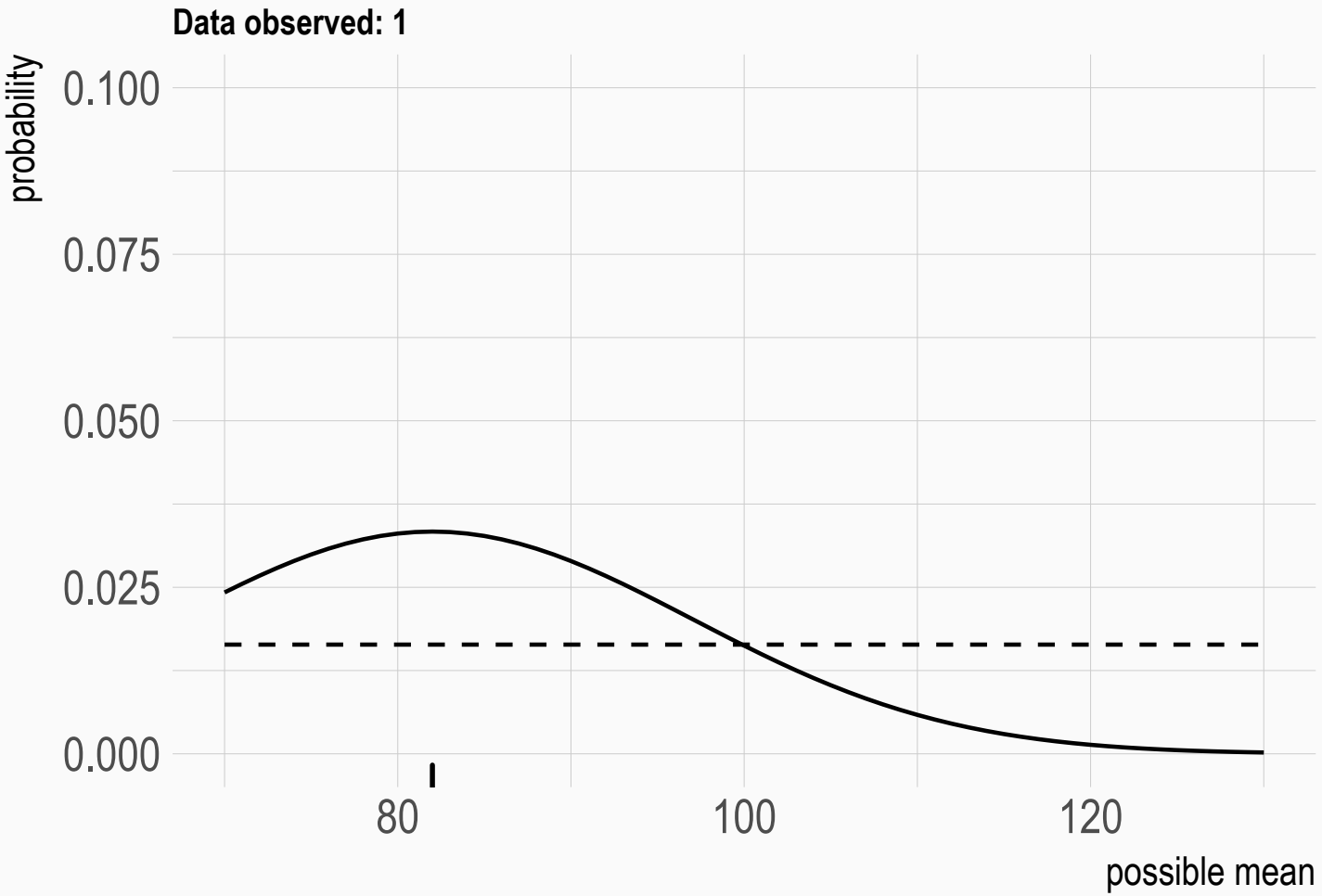
```
df_iq_model$previous <- df_iq_model$prob
likelihoods <- dnorm(iqs[1], df_iq_model$mean, 15)
# numerator of bayes theorem
df_iq_model$prob <- likelihoods * df_iq_model$prob
sum(df_iq_model$prob)
#> [1] 0.01306729
```

We observe one data-point, $y = 82$, and update our prior information using the likelihood of the data at each possible mean.

```
df_iq_model$previous <- df_iq_model$prob
likelihoods <- dnorm(iqs[1], df_iq_model$mean, 15)
# numerator of bayes theorem
df_iq_model$prob <- likelihoods * df_iq_model$prob
sum(df_iq_model$prob)
#> [1] 0.01306729
```

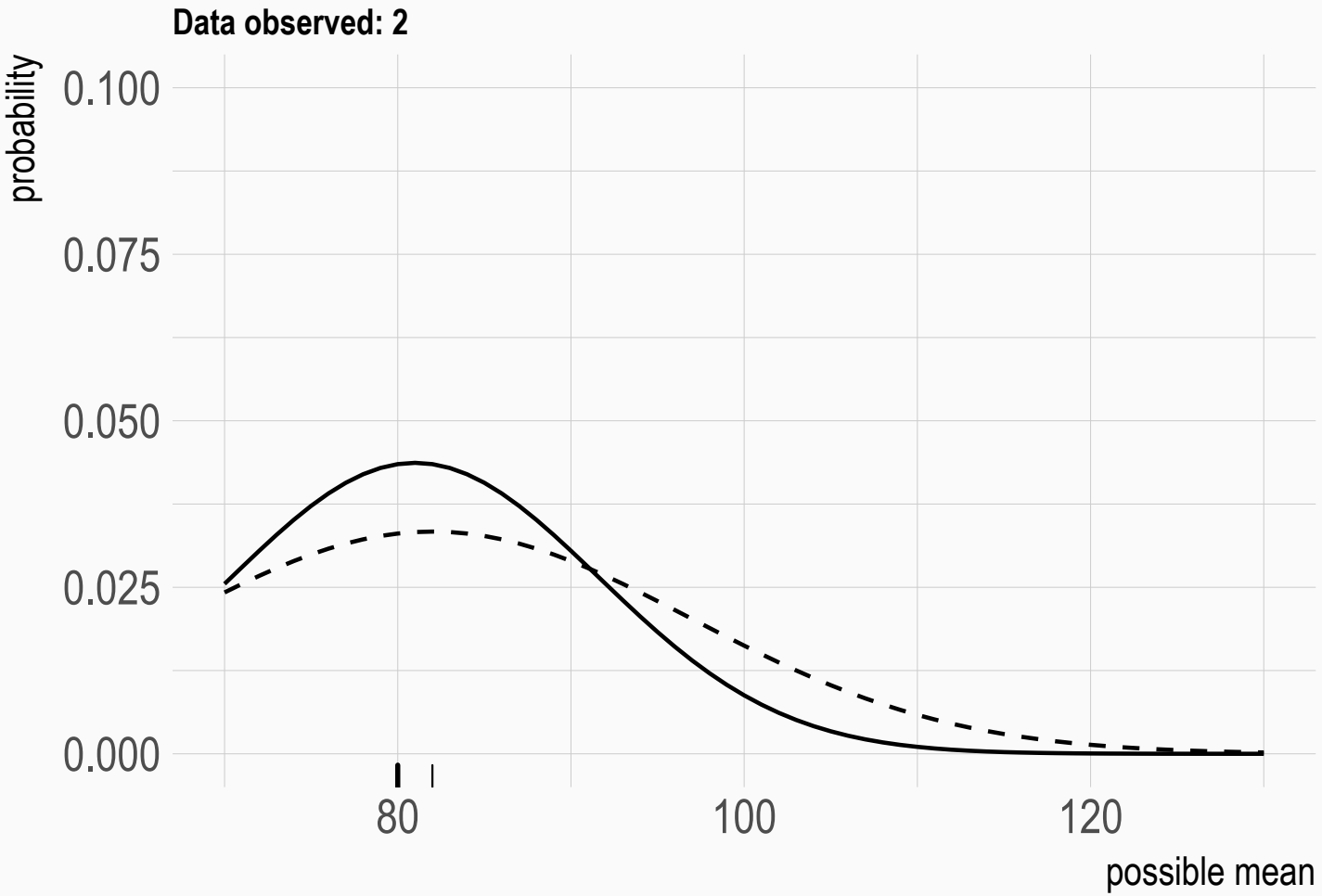
That's not right! We need the *average likelihood* to ensure that the probabilities add up to 1. This is why it's sometimes called a *normalizing constant*.

```
# include denominator of bayes theorem
df_iq_model$prob  <- df_iq_model$prob / sum(df_iq_model$prob)
sum(df_iq_model$prob)
#> [1] 1
```


Data observed: 1
probability
0.100
0.075
0.050
0.025
0.000
80
100
120
possible mean

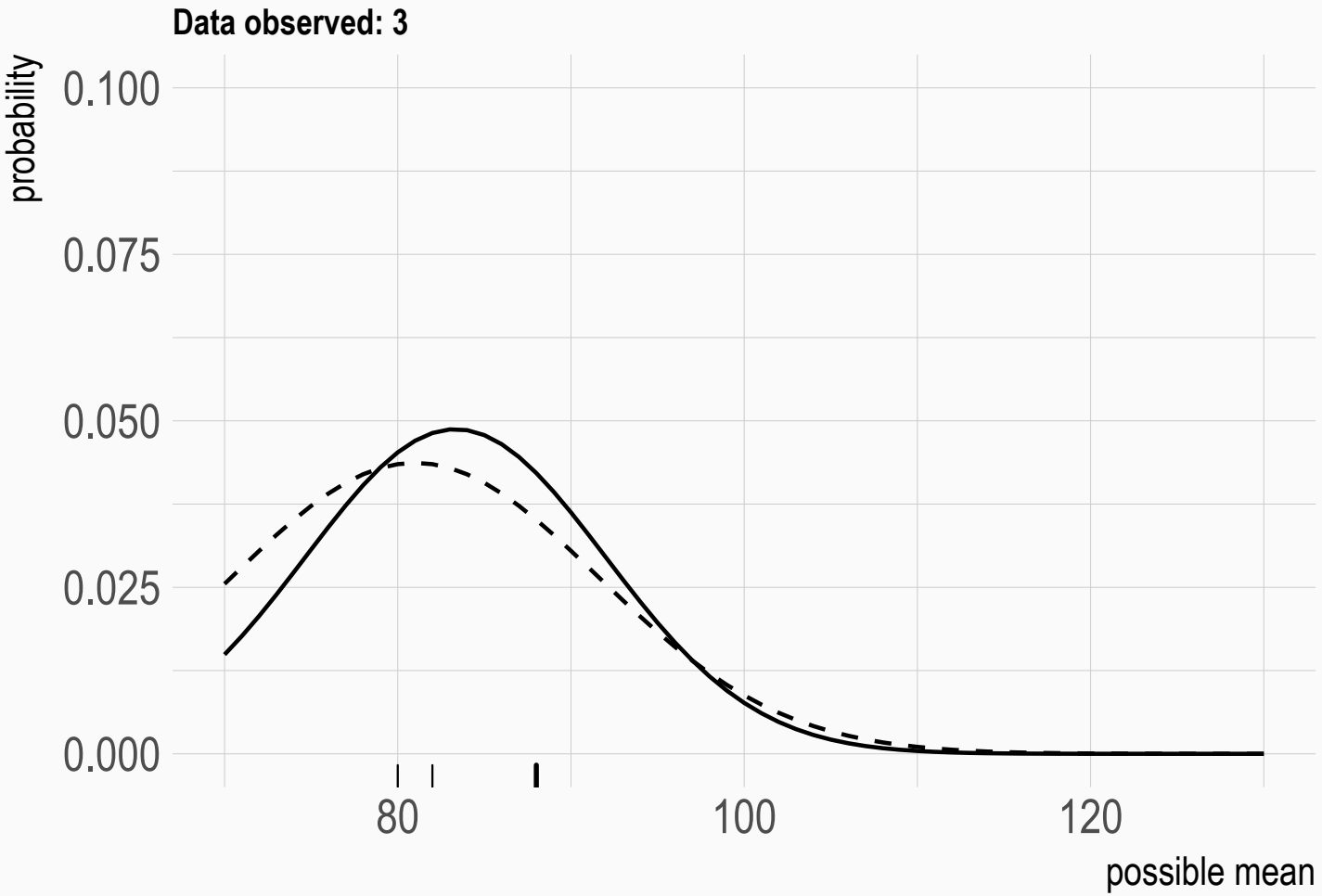
We observe another data-point and update the probability with the likelihood again.

```
df_iq_model$previous <- df_iq_model$prob
likelihoods <- dnorm(iqs[2], df_iq_model$mean, 15)
df_iq_model$prob <- likelihoods * df_iq_model$prob
# normalize
df_iq_model$prob <- df_iq_model$prob / sum(df_iq_model$prob)
df_iq_model
#> # A tibble: 61 × 3
#>     mean       prob   previous
#>    <int>      <dbl>      <dbl>
#> 1     70 0.02551519 0.02422865
#> 2     71 0.02801128 0.02549920
#> 3     72 0.03047942 0.02671736
#> 4     73 0.03287154 0.02786958
#> 5     74 0.03513768 0.02894257
#> 6     75 0.03722765 0.02992359
#> 7     76 0.03909289 0.03080065
#> 8     77 0.04068830 0.03156283
#> 9     78 0.04197406 0.03220045
```


Data observed: 2
probability
0.100
0.075
0.050
0.025
0.000
80
100
120
possible mean

## And one more. . .

```
df_iq_model$previous <- df_iq_model$prob
likelihoods <- dnorm(iqs[3], df_iq_model$mean, 15)
df_iq_model$prob <- likelihoods * df_iq_model$prob
# normalize
df_iq_model$prob <- df_iq_model$prob / sum(df_iq_model$prob)
df_iq_model
#> # A tibble: 61 × 3
#>     mean       prob   previous
#>    <int>      <dbl>      <dbl>
#> 1     70 0.01490139 0.02551519
#> 2     71 0.01768232 0.02801128
#> 3     72 0.02070434 0.03047942
#> 4     73 0.02392174 0.03287154
#> 5     74 0.02727304 0.03513768
#> 6     75 0.03068201 0.03722765
#> 7     76 0.03405991 0.03909289
#> 8     77 0.03730890 0.04068830
#> 9     78 0.04032654 0.04197406
#> 10    79 0.04301093 0.04291726
#> # ... with 51 more rows
```


Data observed: 3
probability
0.100
0.075
0.050
0.025
0.000
80
100
120
possible mean

# An animation of these steps

https://github.com/tjmahr/MadR_RStanARM/blob/master/assets/simple-updating.gif

# Connecting Bayes' theorem to linear regression

# Linear models

I learned stats in this course, so I bet you probably write regression models as a one-liner like:

$$\underbrace{y_i}_{\text{observation}} = \underbrace{\alpha + \beta_1 x_{1i}}_{\text{predicted mean given } x} + \underbrace{\epsilon_i}_{\text{random error}}$$

# Linear models

I learned stats in this course, so I bet you probably write regression models as a one-liner like:

$$\underbrace{y_i}_{\text{observation}} = \underbrace{\alpha + \beta_1 x_{1i}}_{\text{predicted mean given } x} + \underbrace{\epsilon_i}_{\text{random error}}$$

Data generating model: Observation $y_i$ is a draw from a normal distribution centered around a mean.

We estimate the mean with a constant "intercept" term $\alpha$ plus a linear combination of predictor variables (just $x_1$ for now).

Let's re-write the model to make the normal-distribution part clearer. No more one-liner.

$$y_i \sim \text{Normal}(\text{mean} = \mu_i, \text{SD} = \sigma) \qquad \text{[likelihood]}$$

$$\mu_i = \alpha + \beta_1 * x_{1i} \qquad \text{[linear model]}$$

Let's re-write the model to make the normal-distribution part clearer. No more one-liner.

$$y_i \sim \text{Normal}(\text{mean} = \mu_i, \text{SD} = \sigma) \qquad \text{[likelihood]}$$
$$\mu_i = \alpha + \beta_1 * x_{1i} \qquad \text{[linear model]}$$

Observation $y_i$ is a draw from a normal distribution centered around a mean $\mu_i$ with a standard deviation of $\sigma$.

The mean is a constant term $\alpha$ plus a linear combination of predictor variables (just $x_1$ for now).

(These equations describe the same models. It's just a different kind of notation.)

# Weight by height model

Consider a model of weight predicted by height. . .

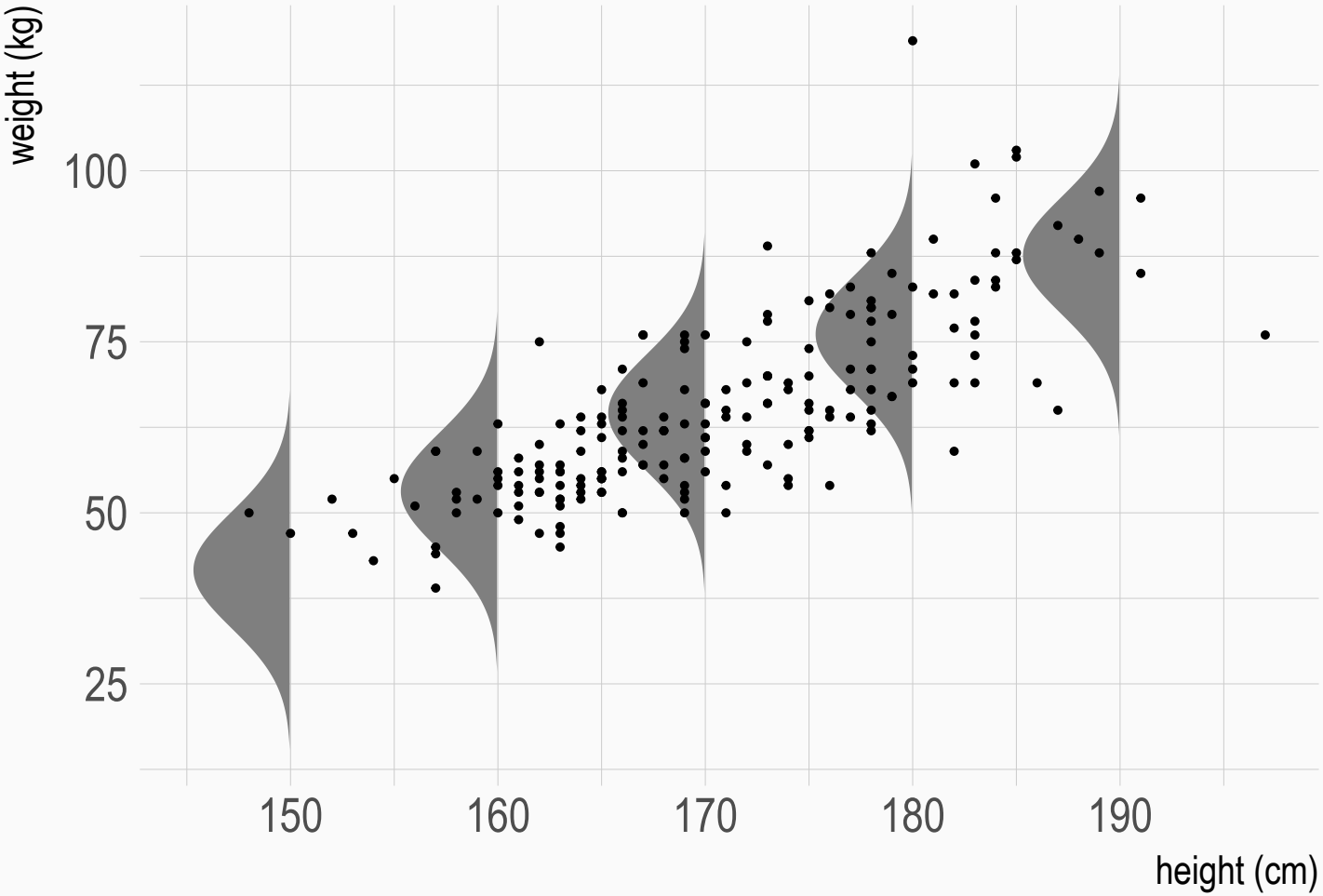



**Figure 8:** It's like a tunnel of bell curves. The center of it moves with $x$.

# Bayesian stats

To make the model Bayesian, we need to give prior distributions to parameters.

The parameters we need to estimate for regression: $\alpha, \beta_1, \sigma$.

# Bayesian stats

To make the model Bayesian, we need to give prior distributions to parameters.

The parameters we need to estimate for regression: $\alpha, \beta_1, \sigma$.

$$
\begin{aligned}
y_i &\sim \text{Normal}(\mu_i, \sigma) && \text{[likelihood]} \\
\mu_i &= \alpha + \beta_1 * x_{1i} && \text{[linear model]} \\
\alpha &\sim \text{Normal}(0, 10) && \text{[prior for } \alpha\text{]} \\
\beta_1 &\sim \text{Normal}(0, 5) && \text{[prior for } \beta_1\text{]} \\
\sigma &\sim \text{HalfCauchy}(0, 5) && \text{[prior for } \sigma\text{]}
\end{aligned}
$$

# What's the point?

- A classical model provides one model of many plausible models of the data. It'll find the parameters that maximize likelihood.
- A Bayesian model is a model of models. We get a *distribution of models* that are consistent with the data.

# But this is where things get difficult!

Parameters we need to estimate: $\alpha, \beta_1, \sigma$

$$\text{posterior} = \frac{\text{likelihood} * \text{prior}}{\text{average likelihood}}$$

# But this is where things get difficult!

Parameters we need to estimate: $\alpha, \beta_1, \sigma$

$$\text{posterior} = \frac{\text{likelihood} * \text{prior}}{\text{average likelihood}}$$

$$P(\alpha, \beta, \sigma \mid x) = \frac{P(x \mid \alpha, \beta, \sigma)\, P(\alpha, \beta, \sigma)}{\iiint P(x \mid \alpha, \beta, \sigma)\, P(\alpha, \beta, \sigma)\, d\alpha\, d\beta\, d\sigma}$$

Things get gnarly. This is the black-box step.

# Good news

We don't perform this integral calculus.

Insead, we rely on Markov-chain Monte Carlo simulation to get samples from the posterior.

Those samples will provide a detailed picture of the posterior.

# Finally, let's fit a model

# An example: Height and Weight by Sex

```
davis
#> # A tibble: 199 × 5
#>       sex weight height repwt repht
#>    <fctr>  <int>  <int> <int> <int>
#> 1       M     77    182    77   180
#> 2       F     58    161    51   159
#> 3       F     53    161    54   158
#> 4       M     68    177    70   175
#> 5       F     59    157    59   155
#> 6       M     76    170    76   165
#> 7       M     76    167    77   165
#> 8       M     69    186    73   180
#> 9       M     71    178    71   175
#> 10      M     65    171    64   170
#> # ... with 189 more rows
```

# Classical linear model

```
# Mean-center height
mean(davis$height)
#> [1] 170.5879
davis$heightC <- davis$height - mean(davis$height)

m <- glm(weight ~ heightC * sex, davis, family = gaussian())
m %>% summary() %>% coef() %>% round(3)
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)   60.558      1.099  55.081    0.000
#> heightC        0.623      0.135   4.626    0.000
#> sexM           7.949      1.710   4.648    0.000
#> heightC:sexM   0.373      0.190   1.964    0.051
```

# What is RStanARM?

Stan: a probabalistic programming language / MCMC sampler

RStanARM: RStan Applied Regression Modeling

- Batteries-included versions of common regression models.
- `glm` -> `stan_glm`, `glmer` -> `stan_glmer`.
- CRAN page is very good! They have lots of detailed vignettes!
- Proper successor to the `arm` package.

```
library(rstanarm)
#> Loading required package: Rcpp
#> rstanarm (Version 2.15.3, packaged: 2017-04-29 06:18:44 UTC)
#> - Do not expect the default priors to remain the same in future rsta
#> Thus, R scripts should specify priors explicitly, even if they are j
#> - For execution on a local, multicore CPU with excess RAM we recomme
#> options(mc.cores = parallel::detectCores())
```

- So. . . hard-code the priors.

# Fit the model

We have to use `stan_glm()`.

- `stan_lm()` uses a different specification of the prior.

By default, it does sampling with 4 MCMC chains. Each "chain" explores the posterior distribution from random starting locations.

- Each chain is 2000 samples, but the first half are warm-up samples.
- Warm-up samples are ignored

```r
stan_model <- stan_glm(
  weight ~ heightC * sex,
  data = davis,
  family = gaussian,
  # RStanARM rescales predictor variables and priors use that scaling
  prior = normal(0, 5),
  prior_intercept = normal(0, 10)
)
#>
#> SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
#>
#> Gradient evaluation took 0 seconds
#> 1000 transitions using 10 leapfrog steps per transition would take 0
#> Adjust your expectations accordingly!
#>
#>
#> Iteration:    1 / 2000 [  0%]  (Warmup)
#> Iteration:  200 / 2000 [ 10%]  (Warmup)
#> Iteration:  400 / 2000 [ 20%]  (Warmup)
#> Iteration:  600 / 2000 [ 30%]  (Warmup)
#> Iteration:  800 / 2000 [ 40%]  (Warmup)
```

# Printing the model

```
stan_model
#> stan_glm
#>  family:  gaussian [identity]
#>  formula: weight ~ heightC * sex
#> ------
#>
#> Estimates:
#>               Median MAD_SD
#> (Intercept)  60.5    1.1
#> heightC       0.6    0.1
#> sexM          8.0    1.7
#> heightC:sexM  0.4    0.2
#> sigma         8.1    0.4
#>
#> Sample avg. posterior predictive
#> distribution of y (X = xbar):
#>          Median MAD_SD
#> mean_PPD 65.3    0.8
#>
```

# One note

Predictors are centered and rescaled internally by rstanarm, so our priors are on the standardized scale.

- `normal(0, 5)` is a distribution of effect sizes with mean 0 and SD 5

See `?rstanarm::priors`, esp. the `autoscale` argument.

```
prior_summary(stan_model)
#> Priors for model 'stan_model'
#> ------
#> Intercept (after predictors centered)
#>  ~ normal(location = 0, scale = 10)
#>      **adjusted scale = 133.43
#>
#> Coefficients
#>  ~ normal(location = [0,0,0], scale = [5,5,5])
#>      **adjusted scale = [ 7.46,66.72,11.81]
#>
```

# Getting a summary from the model

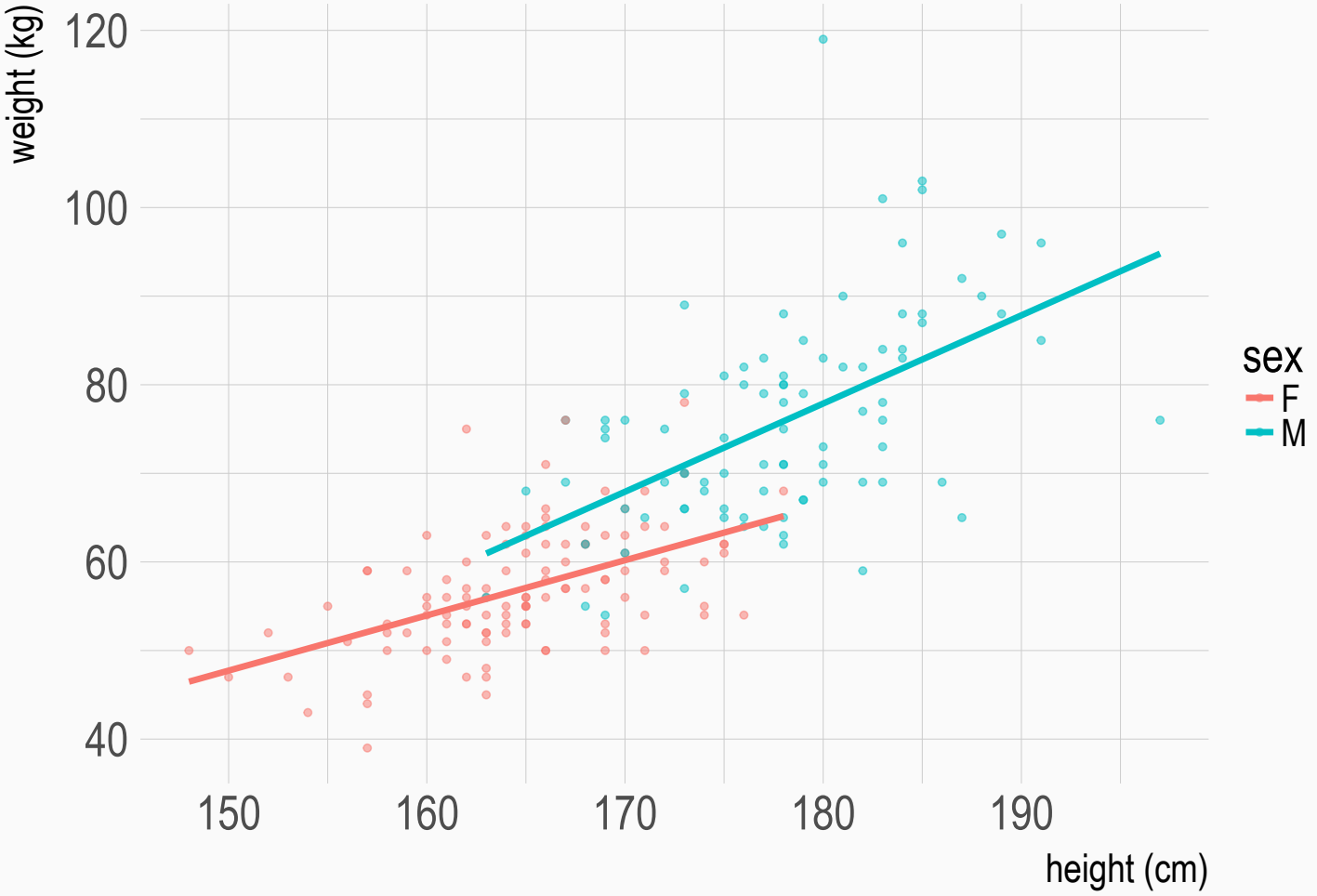
```
summary(stan_model)
#>
#> Model Info:
#>
#>  function:  stan_glm
#>  family:    gaussian [identity]
#>  formula:   weight ~ heightC * sex
#>  algorithm: sampling
#>  priors:    see help('prior_summary')
#>  sample:    4000 (posterior sample size)
#>  num obs:   199
#>
#> Estimates:
#>                mean   sd     2.5%   25%    50%    75%
#> (Intercept)    60.5    1.1   58.4   59.8   60.5   61.3
#> heightC         0.6    0.1    0.4    0.5    0.6    0.7
#> sexM            8.0    1.7    4.6    6.9    8.0    9.1
#> heightC:sexM    0.4    0.2    0.0    0.2    0.4    0.5
#> sigma           8.1    0.4    7.3    7.8    8.1    8.3
```

# Notes on summary()

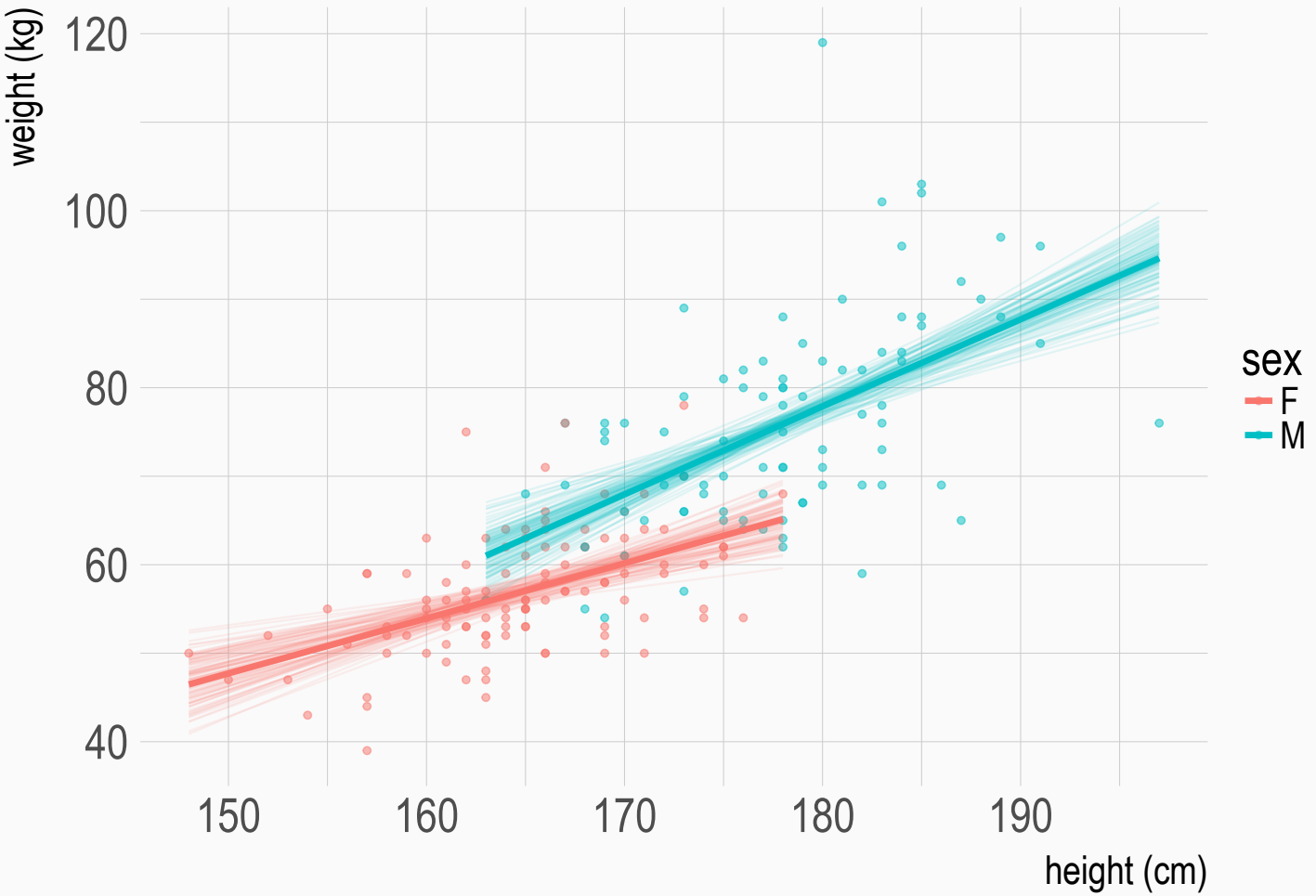
- Split into estimation and diagnostic information
- `mean_PPD` is the predicted value for a completely average observation

# The cut to the chase plot

Here is what classical linear regression does.



# Here is what Bayesian linear regression does



# Inspecting posterior samples

# Looking at the posterior parameter samples

Coerce to a data-frame. Columns are parameters. One row per posterior sample.

```
samples <- stan_model %>% as.data.frame() %>% tbl_df()
samples
#> # A tibble: 4,000 × 5
#>    `(Intercept)`   heightC      sexM `heightC:sexM`
#>            <dbl>     <dbl>     <dbl>          <dbl>
#> 1       62.34053 0.8628468  7.525083   -0.006868722
#> 2       61.32220 0.6687682  7.495325    0.343557108
#> 3       61.00733 0.8070026  9.958593   -0.067616341
#> 4       60.83916 0.7874834 10.012439   -0.111454059
#> 5       59.00949 0.6879592 12.568365   -0.094468079
#> 6       63.30747 0.7219255  2.885327    0.603136203
#> 7       63.32419 0.7605835  3.697716    0.536910207
#> 8       61.06933 0.5777876 10.398769    0.205390488
#> 9       60.35789 0.5115736  9.610737    0.393665653
#> 10      59.50617 0.4236184  9.760122    0.341698454
#> # ... with 3,990 more rows, and 1 more variables:
```

# We have a distribution

Any stats that can describe a distribution can describe the model's parameters now. Mean, median, skew, quantiles, etc.

# Looking at the posterior parameter samples

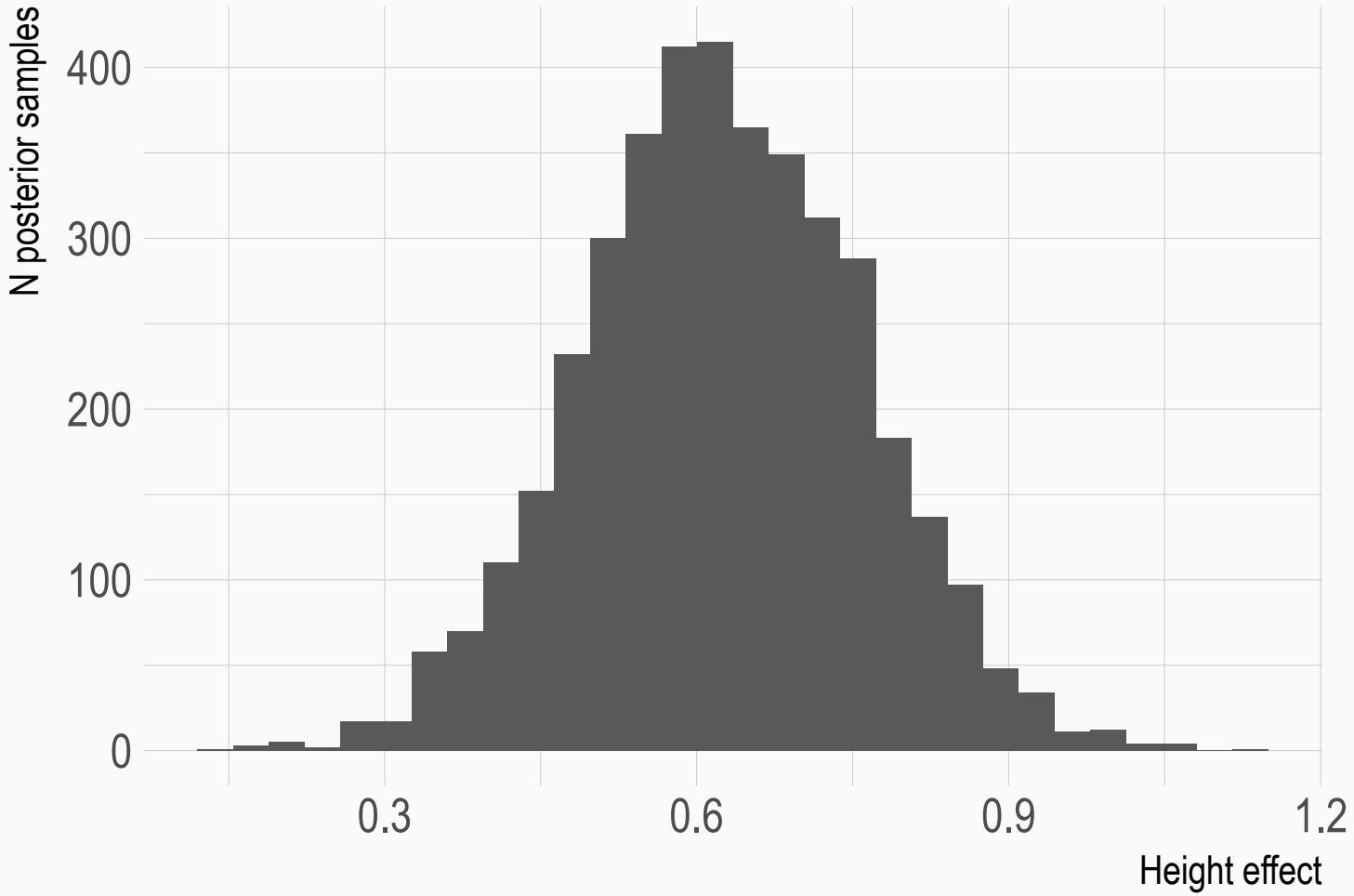



**Figure 9:** Histogram of height effect.

# Quantiles are post-data probabilities

If we believe there is a "true" value for a parameter, there is 90% probability that this "true" value is in the 90% interval, given our model, prior information, and the data.

The 90% interval contains the middle 90% of the parameter values.

There is a 5% chance, says the model, the height parameter that generated the data is below the 5% quantile.

```
posterior_interval(stan_model)
#>                          5%        95%
#> (Intercept)  58.77478527 62.2842043
#> heightC       0.40448188  0.8439221
#> sexM          5.13785584 10.7705237
#> heightC:sexM  0.05531133  0.6781179
#> sigma         7.41739002  8.7922145
```

# Live demo

```
# This is where I toured launch_shinystan(stan_model)
# and did some other stuff.
```

# My experience with this framework

# Benefits

The models provide intuitive results.

- When we misinterpret $p$-values or confidence intervals, we usually are interpreting them in a Bayesian way.
- . . . so Bayesian uncertainty intervals are what we want from confidence intervals.

Bayesian models quantify uncertainty.

- Basically, if a classical model can estimate or predict something about the data, the Bayesian model can estimate a distribution for that thing too.
- Bayesian models are generative, and the posterior predictive distribution (which simulates fake data using the model) is a useful tool.

Bayesian models incorporate prior information.

- That information can be weak, moderate or strong.
- I don't say "prior beliefs" because that sounds too subjective.
- All models make assumptions and build on prior information, and priors make that information explicit.

Bayesian models are flexible.

- This course captures a bag of tricks (*t*-tests, ANOVA, ANCOVA, mixed effects) under a general framework: it's all regression.
- Bayesian regression incorporates even more tricks (missing data imputation, measurement error models, robust error models) into the framework.

Bayesian models have computational benefits.

- Multilevel models with lots of random effects probably won't converge.
- But some weak prior information will nudge the models in the right direction and make the models work.

# Downsides

It's different.

- People are really used to signficance testing and *p*-values, so you have to do more hand-holding when explaining results.
- You don't get to say *significant* anymore. (I use *plausible* and *credible*.)
- People have misconceptions about subjectivism and bias.

More work before and after modeling.

- You need to specify priors for your parameters.
- Your model is a distribution, so you have to do a bit more work wrangling the data.

It takes longer.

- Classical models solve an optimization problem and provide a single set of parameter estimates.
- MCMC sampling explores the space of parameter values and provides thousands of parameter estimates.
- It can take a few hours to fit a complicated multilevel model.

It's not a cure-all. There are still insecurities.

- It's statistics and people can still misunderstand the methods and models.
- A motivated $p$-hacker can still exploit Bayes factors, which is why I won't discuss them.

# Learn more

These are some older slides on good resources for learning about Bayesian statistics.

`https://cdn.rawgit.com/tjmahr/MadR_RStanARM/master/04-learning-more-rpubs.html`