Become a Bayesian in 3 simple steps

Josh Jackson

Reassurance, before we get to the three steps

- Not drastically different!
- You get to keep everything you like
- Your models stay the same!!

GLM

- Our good friend that gives us 99% of the models psychologists use (general(ized) linear model), is exactly the same

$$Y = b_0 + b_1 X + e$$

- No need to think about setting up new t-test, ANOVAS, regressions, etc. ALL THE SAME.

A working mental model

What are Bayesian models?

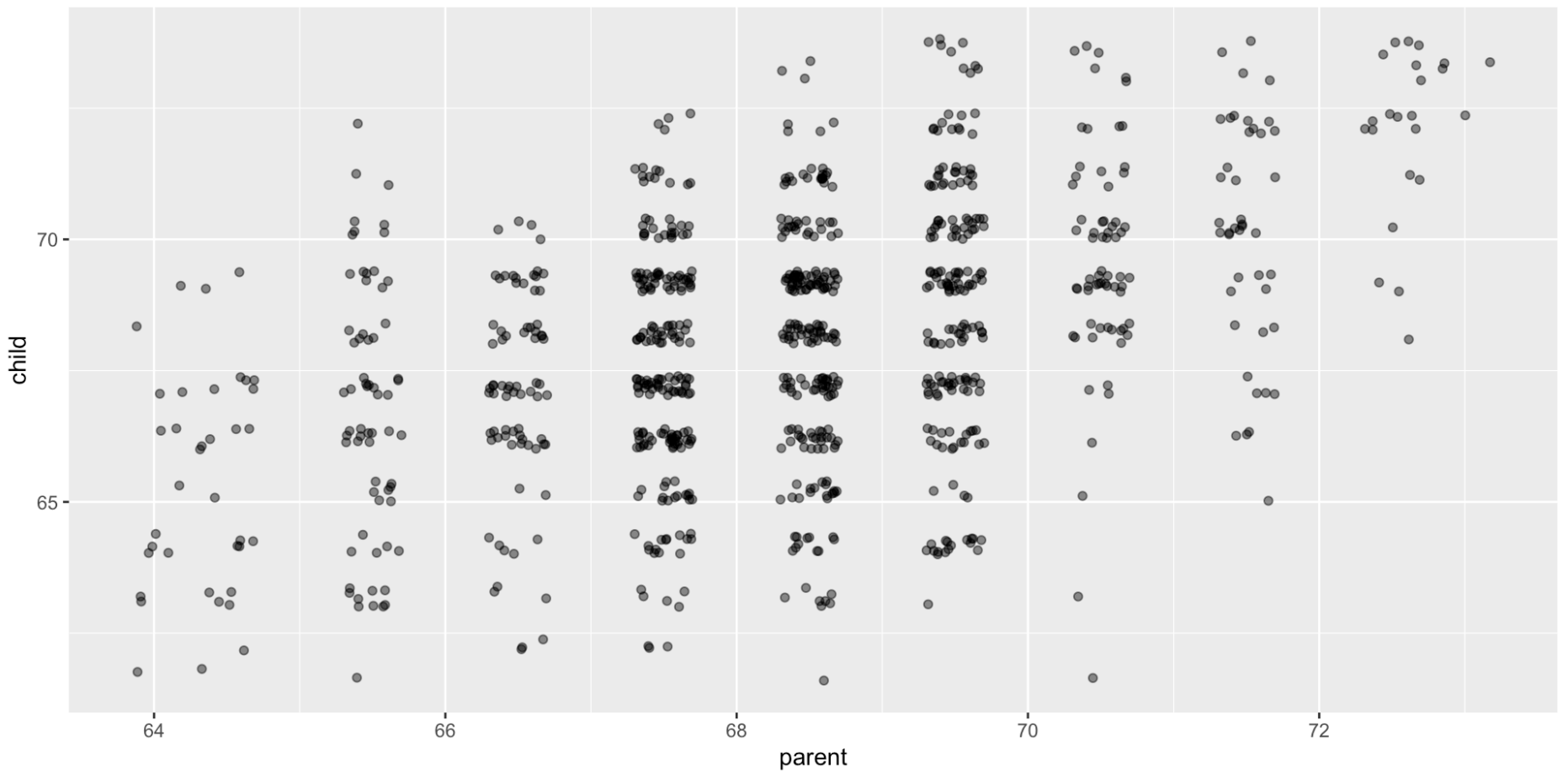
1. “Normal” regression with a different algorithm.
2. Results that represent a distribution rather than a point estimate and some uncertainty.
3. Priors that incorporate existing knowledge.

1. Be comfortable with a different estimation algorithm

- What do you mean by estimation algorithm?
- OLS i.e. $min \sum (Y_i - \hat{Y})^2$
- Fun fact, R uses QR decomposition, Newton Raphson, Fisher Scoring, SVR, etc – not this equation.
- Another fun fact, more advanced stats use an even different algorithm (e.g., maximum likelihood)

Standard way

- Code



▶ Code

```
Call:
lm(formula = child ~ parent, data = galton.data)

Residuals:
    Min      1Q  Median      3Q     Max
-7.8050 -1.3661  0.0487  1.6339  5.9264

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 23.94153    2.81088   8.517   <2e-16 ***
parent       0.64629    0.04114  15.711   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Fisher Scoring

- Code

```
Call:
glm(formula = child ~ parent, family = gaussian, data = galton.data)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 23.94153    2.81088   8.517   <2e-16 ***
parent       0.64629    0.04114  15.711   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for gaussian family taken to be 5.011094)

    Null deviance: 5877.2  on 927  degrees of freedom
Residual deviance: 4640.3  on 926  degrees of freedom
```

Maximum likelihood

```
lavaan 0.6.17 ended normally after 1 iteration

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                         3

  Number of observations                           928

Model Test User Model:

  Test statistic                                 0.000
  Degrees of freedom                                 0

Parameter Estimates:
```

Bayesian way

▶ Code

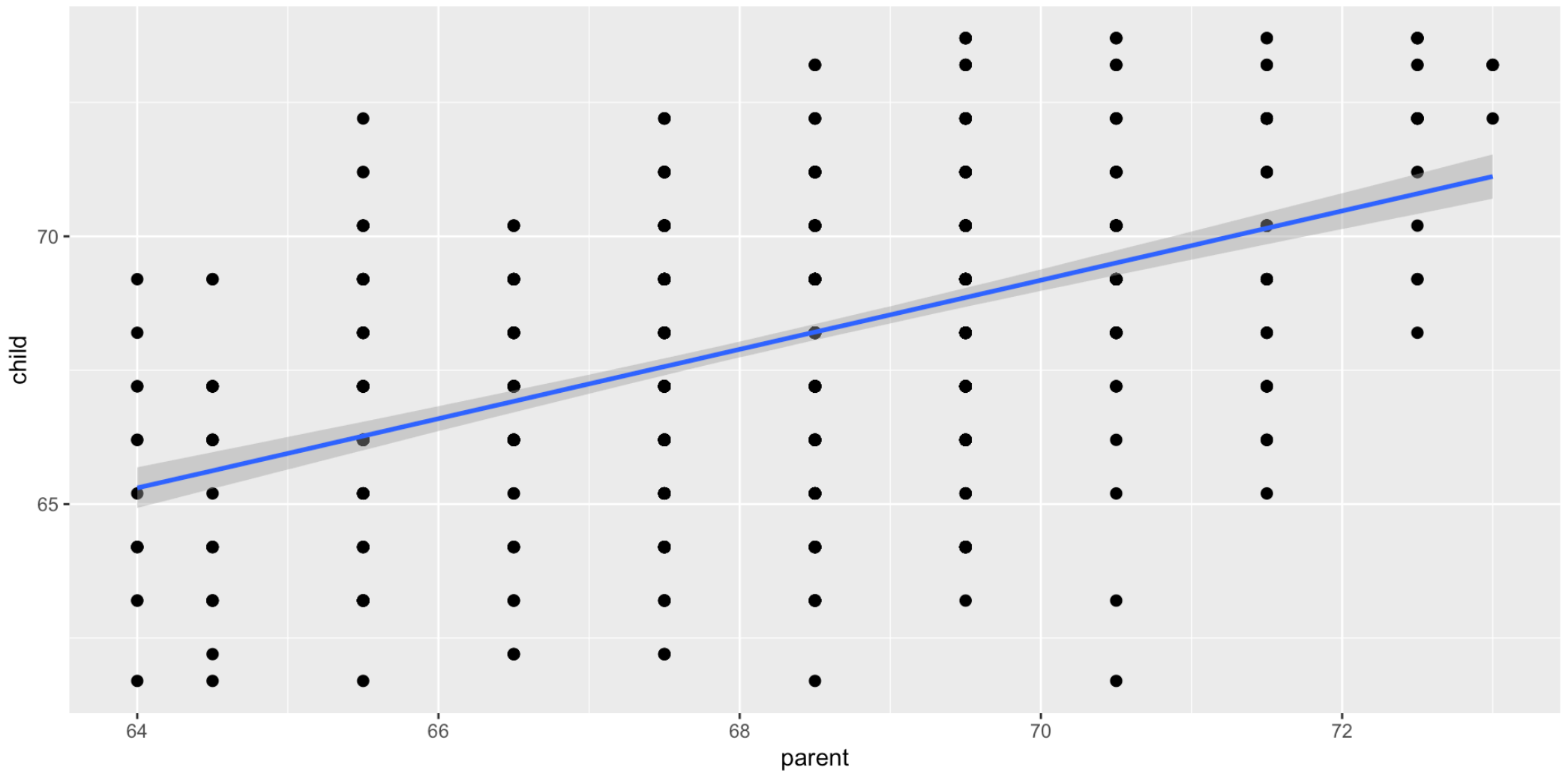
```
1  summary(fit.1.bayesian)
```

```
 Family: gaussian
  Links: mu = identity; sigma = identity
Formula: child ~ parent
   Data: galton.data (Number of observations: 928)
  Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
         total post-warmup draws = 4000

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept    23.98      2.83    18.46    29.71 1.00     3706     2815
parent        0.65      0.04     0.56     0.73 1.00     3710     2971

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     2.24      0.05     2.15     2.34 1.00     4233     2967
```

▶ Code



Step 1 is easy

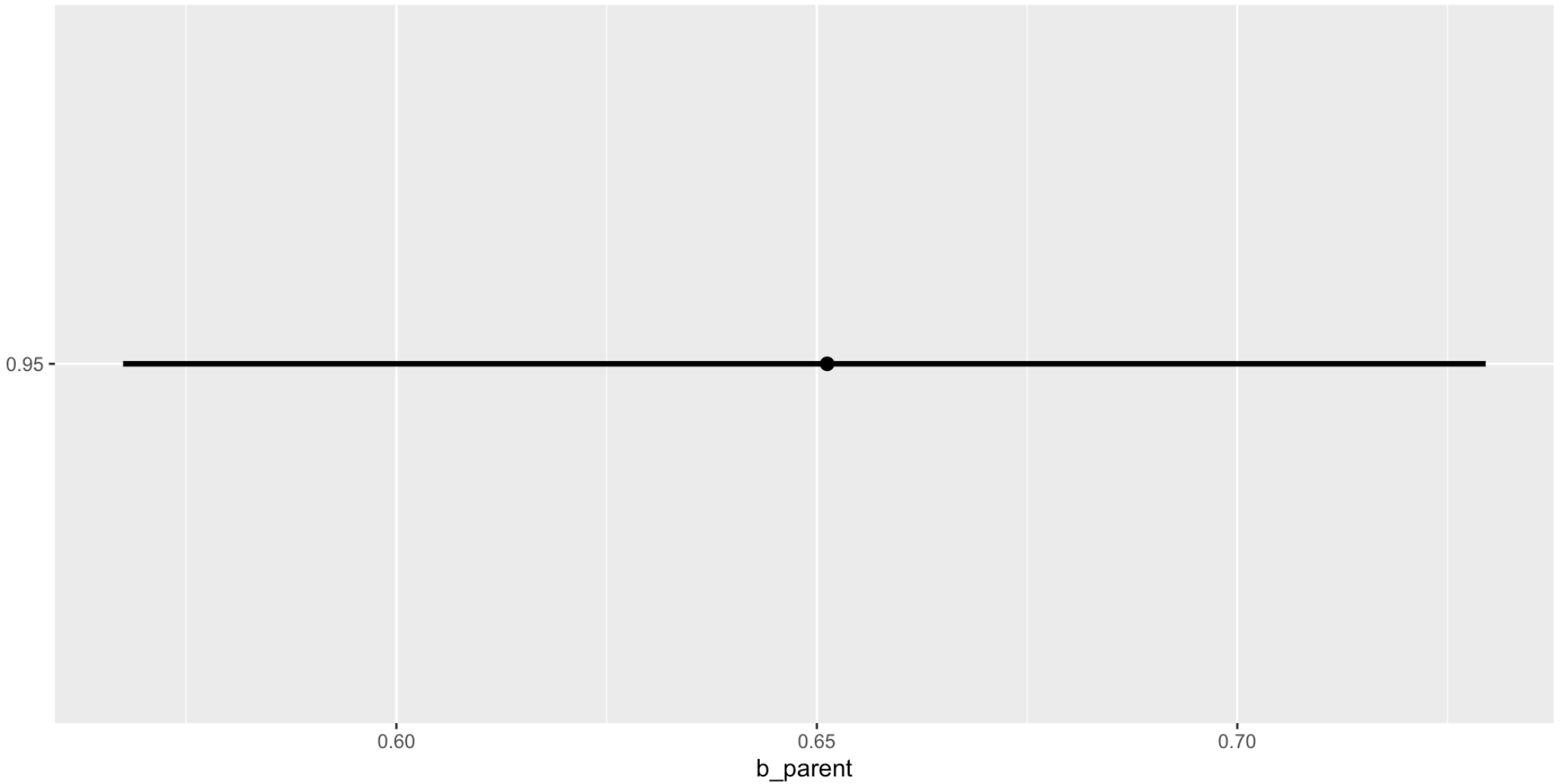
- Bayes gives you basically the same results
- So why use it? Many reasons, but the most direct is manipulating, visualizing, and extrapolating from results

2. Think of results in terms of distributions

- What are results?

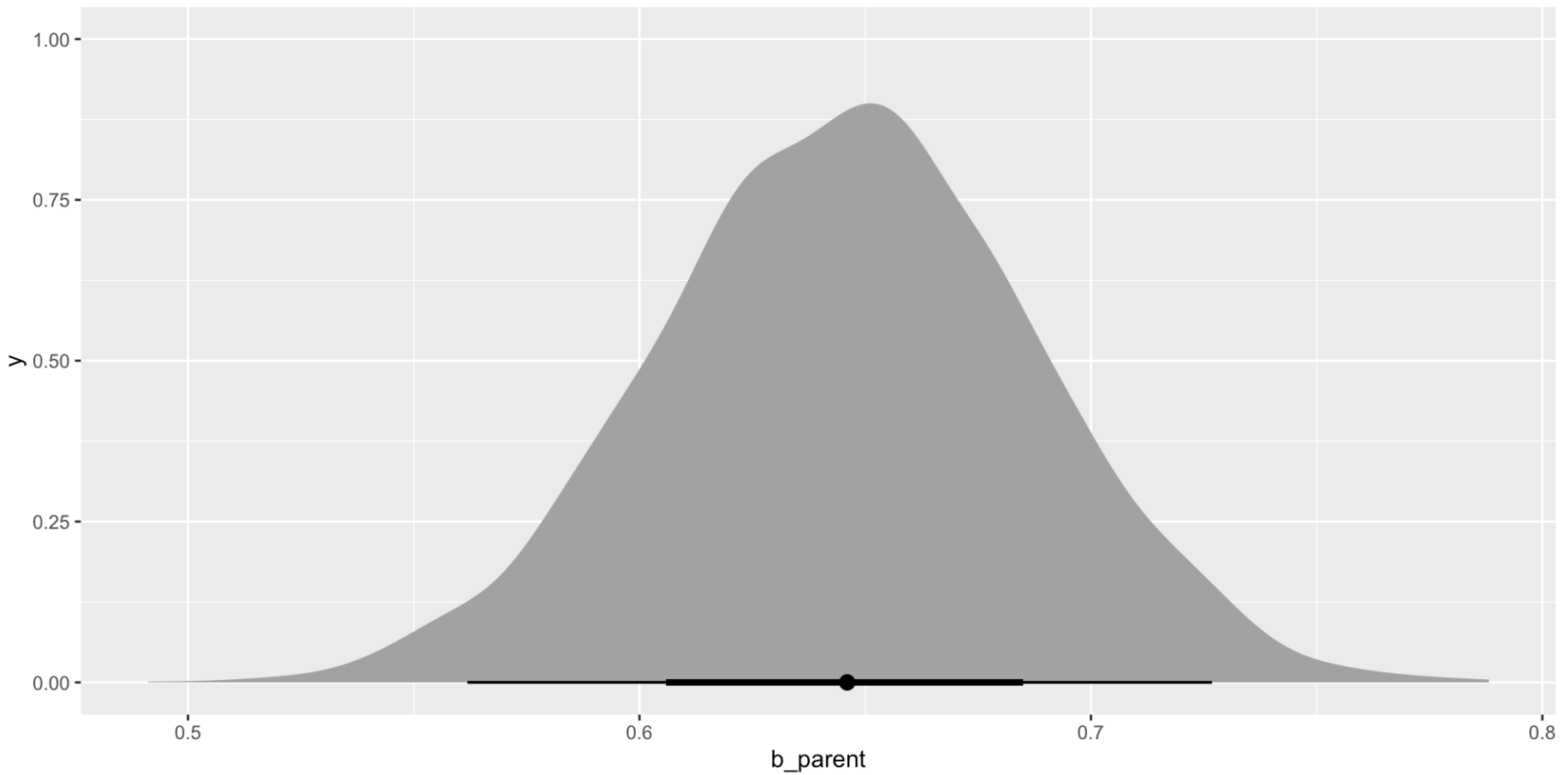
- Estimate and an SE
- Indicates a “best guess” ie mean/median/mode and the imprecision related to it
- If this guess is far away from zero (and imprecision not large), then it is significant
- We know that if we repeated this again we won’t get the same answer (estimate), but likely in between our CIs
- How do we convey the “best guess?”

▶ Code



- The problem is they obscure information

▶ Code



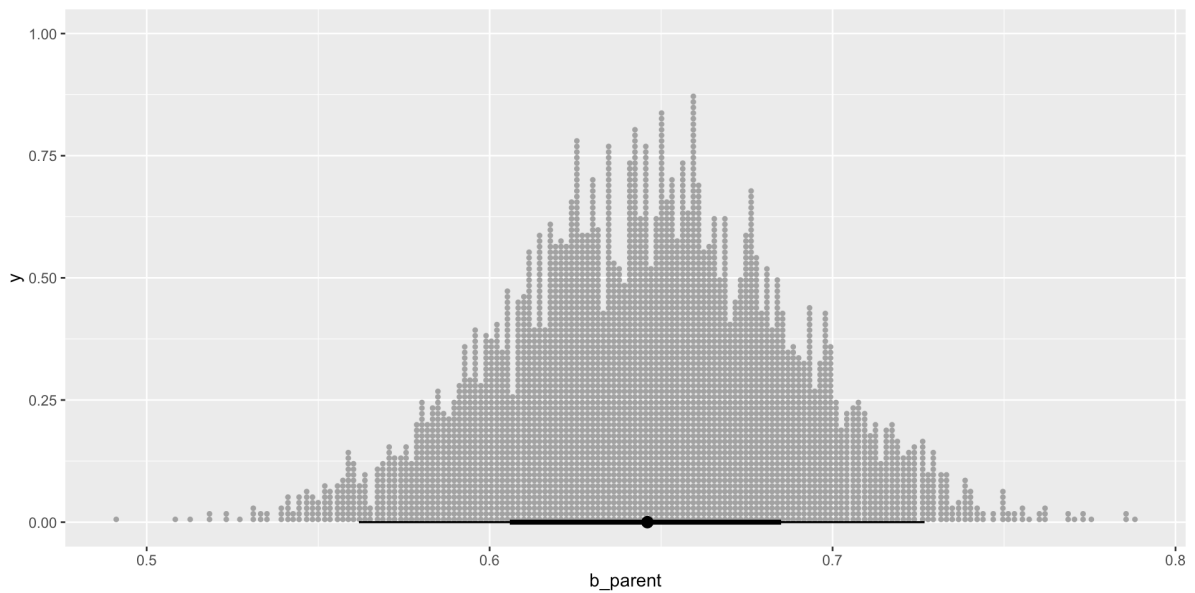
Posterior distribution (ie results)

- Is made up of a series of educated guesses (via our algorithm), each of which is consistent with the data.
- In aggregate, these guesses provide us not with a best guess and an SD (as with Maximum Likelihood), but a more complete sense of each parameter we are trying to estimate.
- We can assume this distribution (typically normal) with standard estimation, but with bayes it can be flexible!

Posterior distribution (ie results)

Is made of up of a series of educated guesses. Each dot represents a particular guess. Guesses that occur more often are considered more likely.

▶ Code



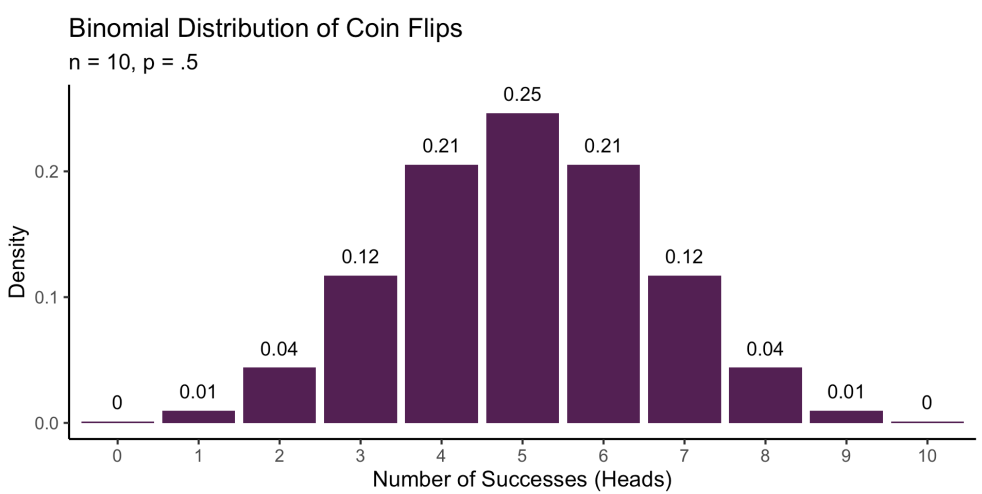
How does the algorithm work?

- Played a role in developing the thermonuclear bomb with one of the earliest computers. Published in 1953 but ignored within stats b/c it was published within a physics/chemistry journal. Took about until 1990 for desktop computers to run fast enough to do at home.
- Many variants, but the general idea is a) propose an estimate value + noise N(0, σ) then b) see how "likely" the data is given the estimate, c) based on some criteria (better than worse that some value) either accept or reject the estimate and d) repeat

What do you mean by likely?

You've done this before last semester. Three parameters in a binomial distribution (# successes, # of trials, probability of success). Often you would fix #trials and probability of success to see what # successes are most/least likely.

▶ Code



- But we often don’t know what P is. That is the parameter we want to estimate. But we collected data! So we can look at what p is consistent (or not) with our data (2 successes in 10 trials).
- This is basically what our current ML algorithms do.

▶ Code

- The Bayesian (MCMC) algorithm tries out a bunch of parameter values. The one's that are *more likely* will appear more often.
- What do I mean "appear" more often. The algorithm lands on that just as our coin flipping example finds .2 to be most likely.

▶ Code

Our posterior is literally made up of educated guesses by the algorithm

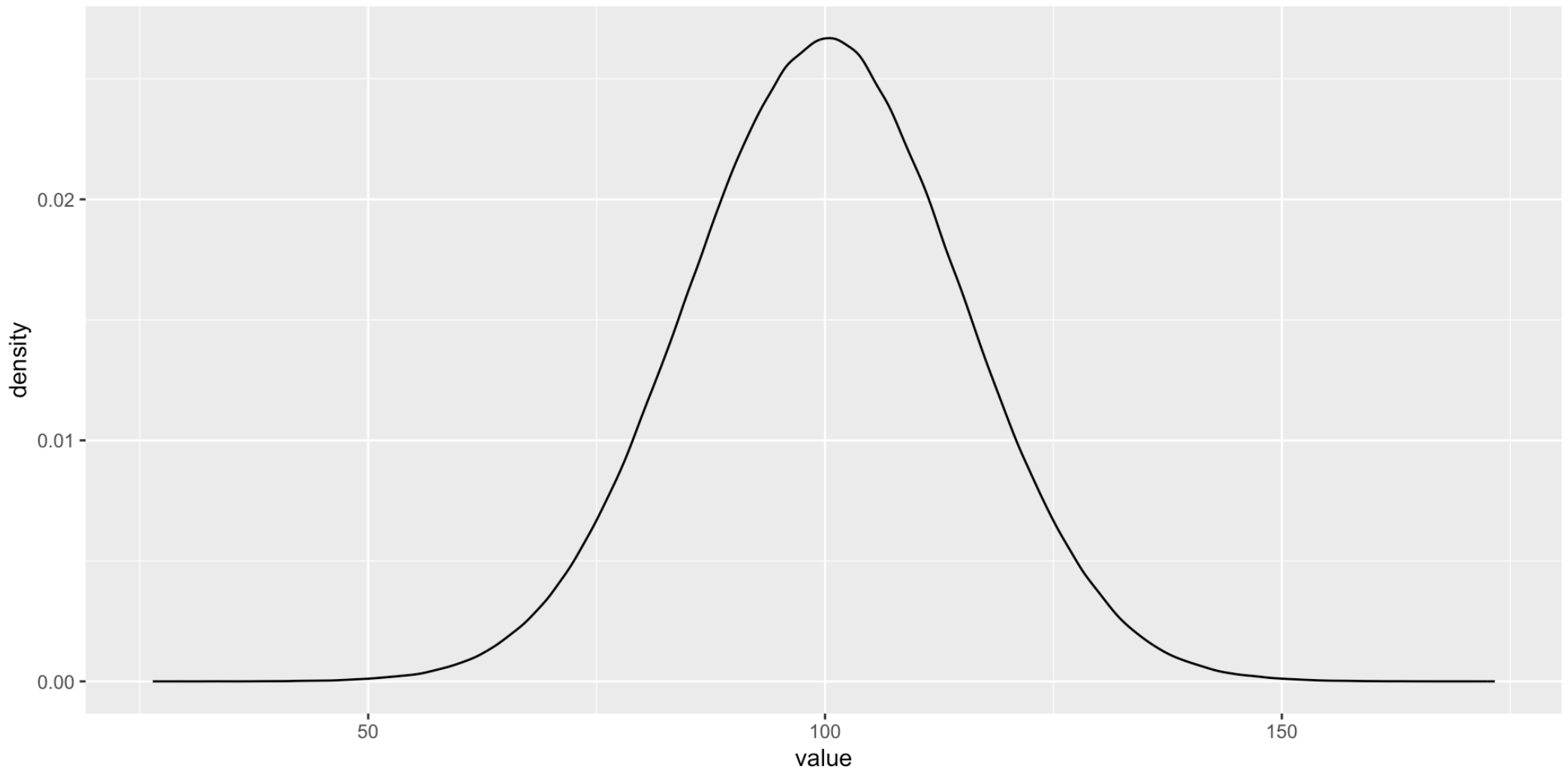
```
1 tidy_draws(fit.1.bayesian)
```

```
# A tibble: 4,000 × 13
   .chain .iteration .draw b_Intercept b_parent sigma   lp__ accept_stat__
    <int>      <int> <int>       <dbl>    <dbl> <dbl>  <dbl>         <dbl>
 1      1          1     1        25.2    0.630  2.35 -2070.         0.595
 2      1          2     2        17.8    0.737  2.22 -2069.         0.972
 3      1          3     3        30.2    0.555  2.25 -2069.         0.992
 4      1          4     4        29.8    0.562  2.17 -2070.         0.942
 5      1          5     5        21.7    0.678  2.14 -2069.         0.916
 6      1          6     6        24.0    0.643  2.31 -2068.         0.919
 7      1          7     7        21.2    0.687  2.16 -2069.         0.981
 8      1          8     8        29.2    0.569  2.29 -2069.         0.962
 9      1          9     9        29.1    0.571  2.25 -2068.         1
10      1         10    10        18.4    0.727  2.25 -2069.         0.934
# ℹ 3,990 more rows
# ℹ 5 more variables: stepsize__ <dbl>, treedepth__ <dbl>, n_leapfrog__ <dbl>,
```

More intuituion

- Think of the algorithm as picking out marbles from a sack, with replacement, to figure out the distribution of colors.
- Or us doing `rnorm` with me hiding what the mean and SD are, but then figuring out what the mean and SD are through counting the samples.

▶ Code

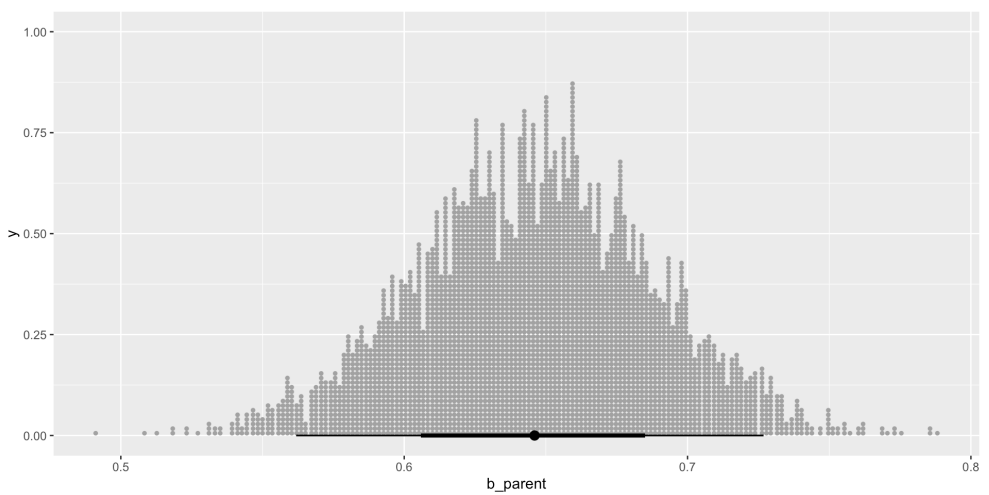


Bayesian analysis is just counting

Visualizing uncertainty

Our posterior (ie different educated guesses at a the correct parameters; distribution of plausible values) is highlighting: that there is no ONE result, that there are many possible results that are consistent with the data.

▶ Code



Some positives of focusing on uncertainty

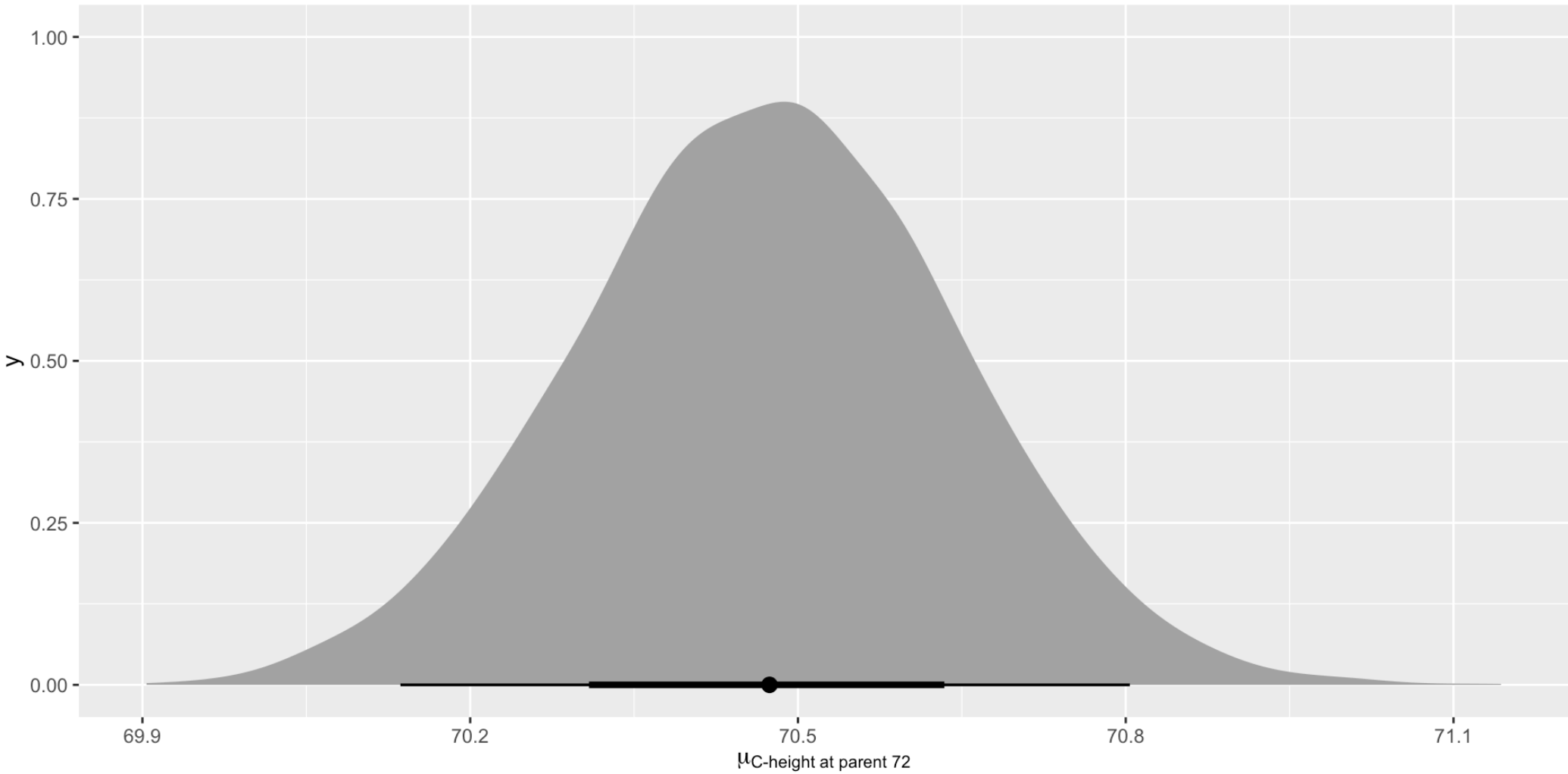
1. Do not need to assume normal or multivariate normal. Uncertainty does not need to be even tailed.
2. Differences (say across groups) in uncertainty is allowed. Do not need to assume groups have same standard errors. One can better account for and/or probe situations where a certain group has a lot or little variability.
3. Easy to calculate uncertainty

CIs around a particular value

With your current knowledge, calculate a 95% CI around parent = 72 inches, to tell you what is possible for the sample mean at that hight.

$$\hat{Y} \pm t_{critical} * se_{residual} * \sqrt{\frac{1}{n} + \frac{(X - \bar{X})^2}{(n-1)s_X^2}}$$

▶ Code

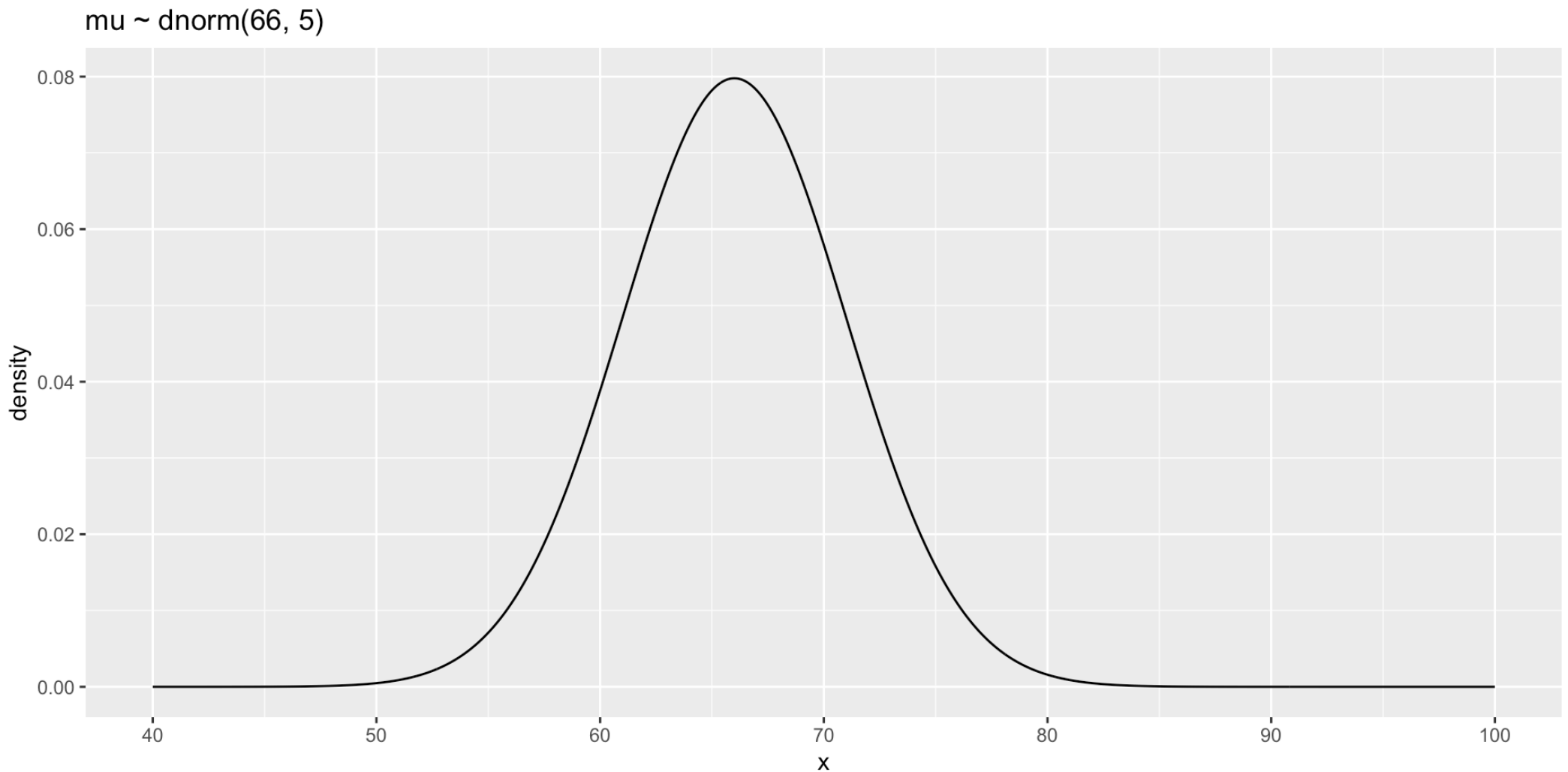


3. Integrating prior knowledge

- Priors insert knowledge you have outside of your data into your model
- This can seem “subjective” as opposed to the more “objective” way of letting the data speak.

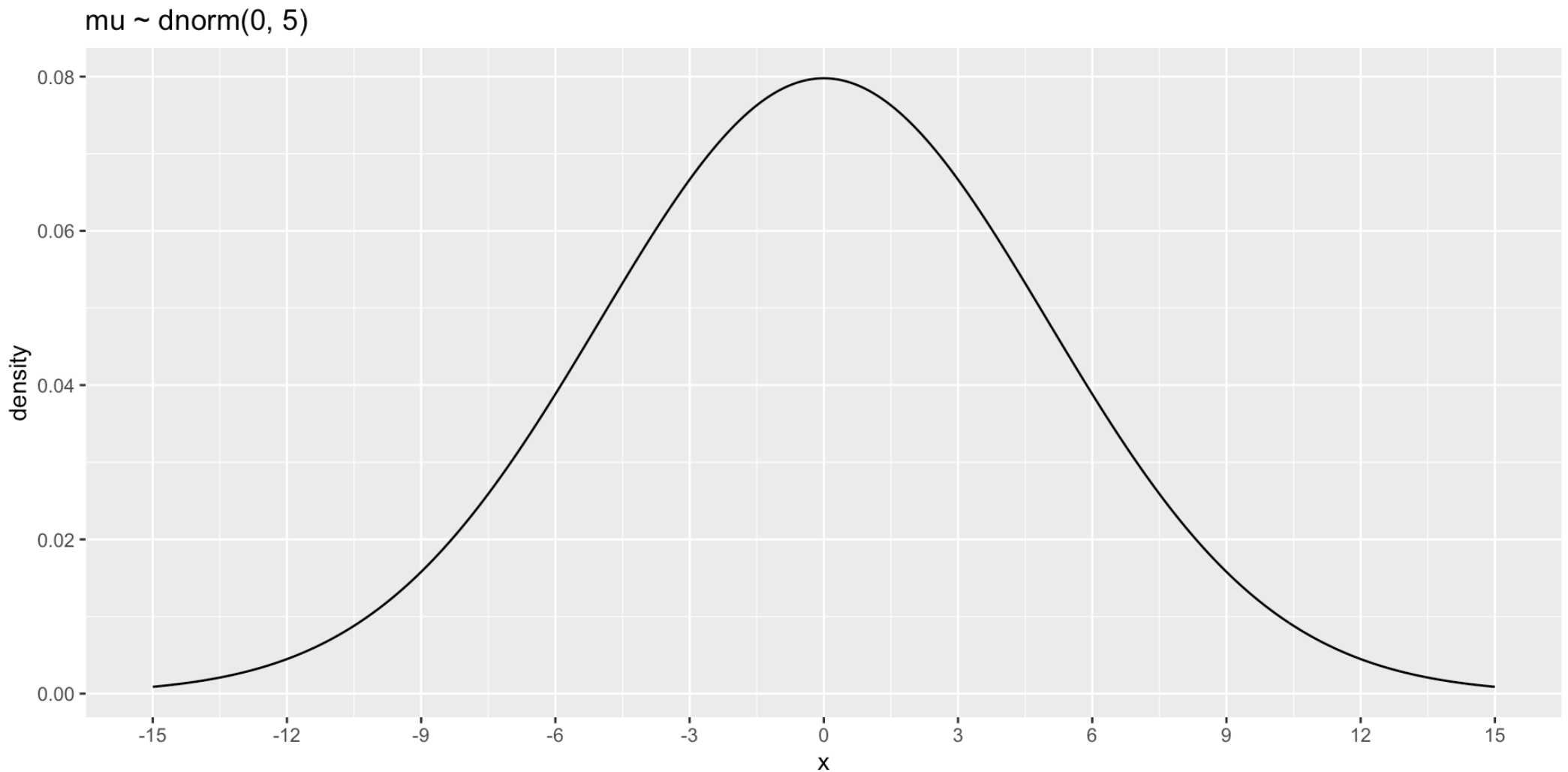
- Take our height example where we are fitting
$$Child = b_0 + b_1 Parent + e$$
- We need to put priors on each parameter we want to estimate, here b_0 & b_1 (and e).
- b_0 is the intercept and reflect average child height when parent height is centered.
- We know, roughly, what average height of adults are so we can create a distribution, say ~N(66 (5.5 ft), 5). That means we are pretty sure (95%) the average height is between ~4’8 and 6`4

▶ Code



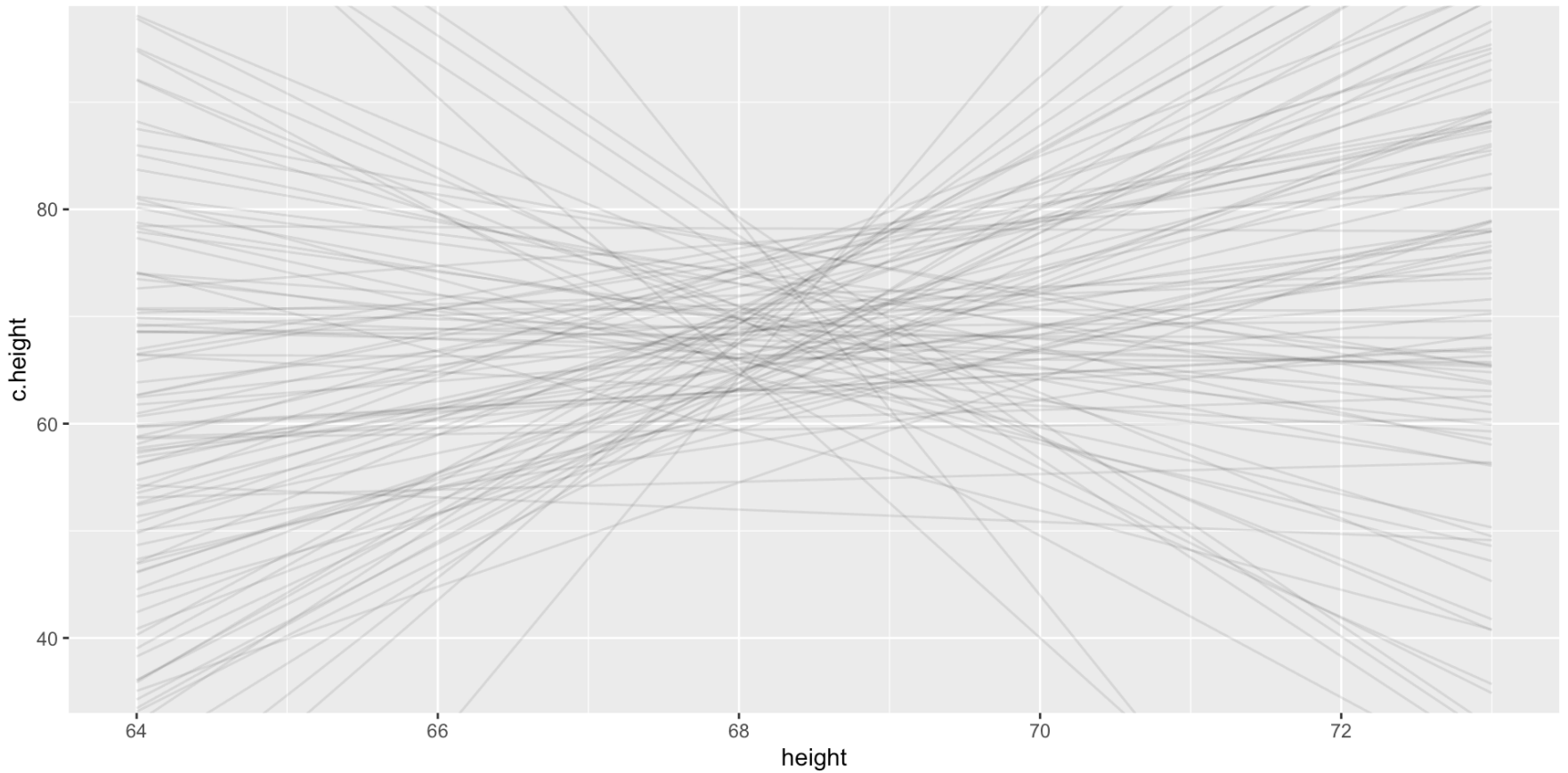
- We could argue that the b_1 parameter (which indexes the strength of association between parent and child height) is positive. But we don't want to stack the deck.
- Let's center it around zero, saying that the most plausible estimate is no association, but that we are willing to entertain some strong effects in either direction.

▶ Code



Okay so what does this mean?

It means, *BEFORE WE SEE THE DATA* we are comfortable with different regression lines.



Okay so why is this important?

- A model that makes impossible predictions prior to seeing the data isn't too useful. Why waste the effort? We often know what values are likely, given what we know about effect sizes
- This is exactly what we do with standard "frequentist" methods. They have implicit priors such that all values, from negative infinity to positive infinity are equally likely.
- If we use priors from a uniform distribution we will get the EXACT same results as a frequentist method.

Tying it together

1. Be comfortable with a different estimation algorithm
2. Think of results in terms of distributions
3. Be comfortable integrating prior knowledge

$$p(\theta|data) \propto \frac{p(data|\theta) \times p(\theta)}{p(data)}$$

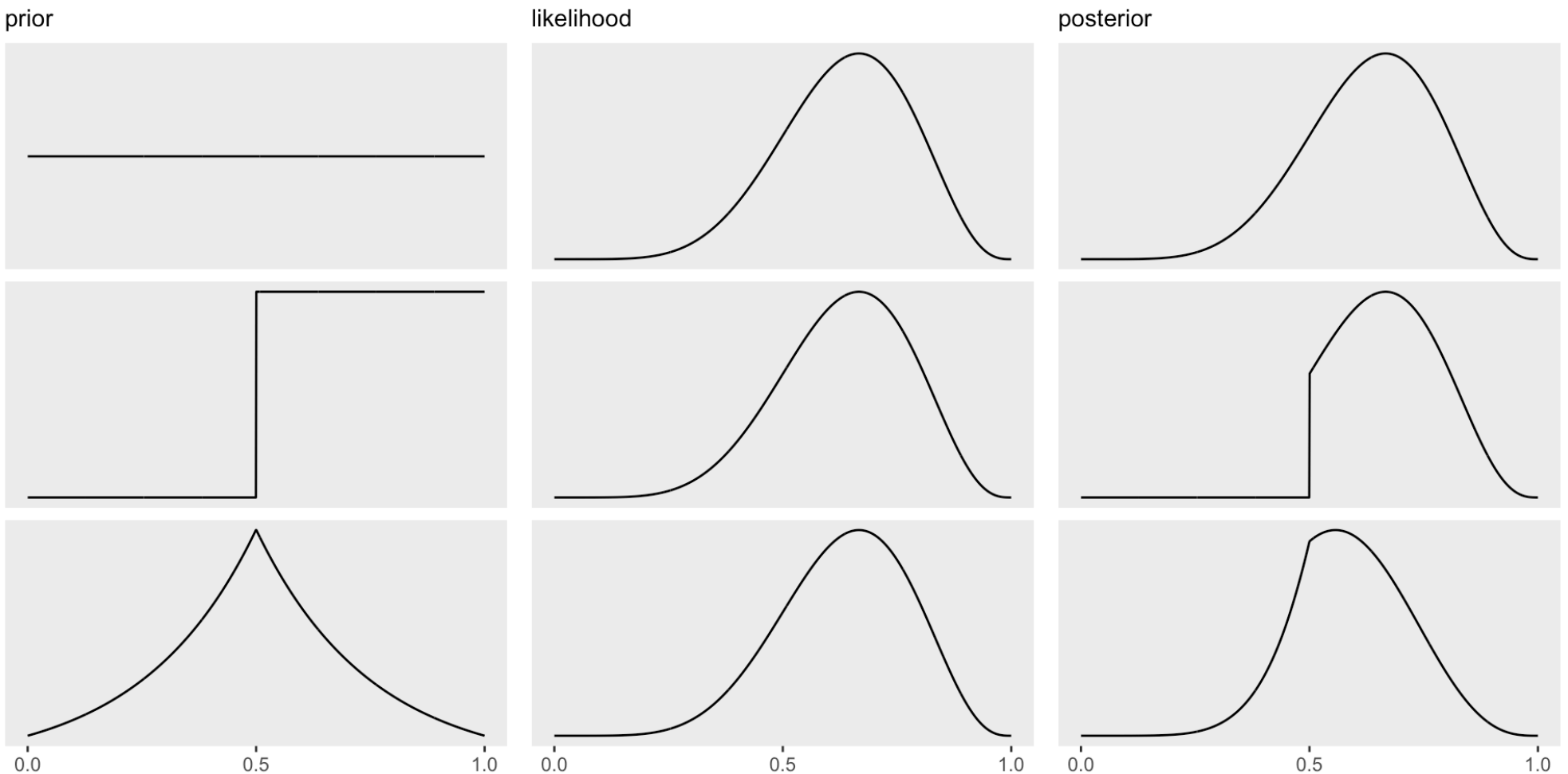
P(θ|data) is the posterior probability.

P(θ) is the prior probability.

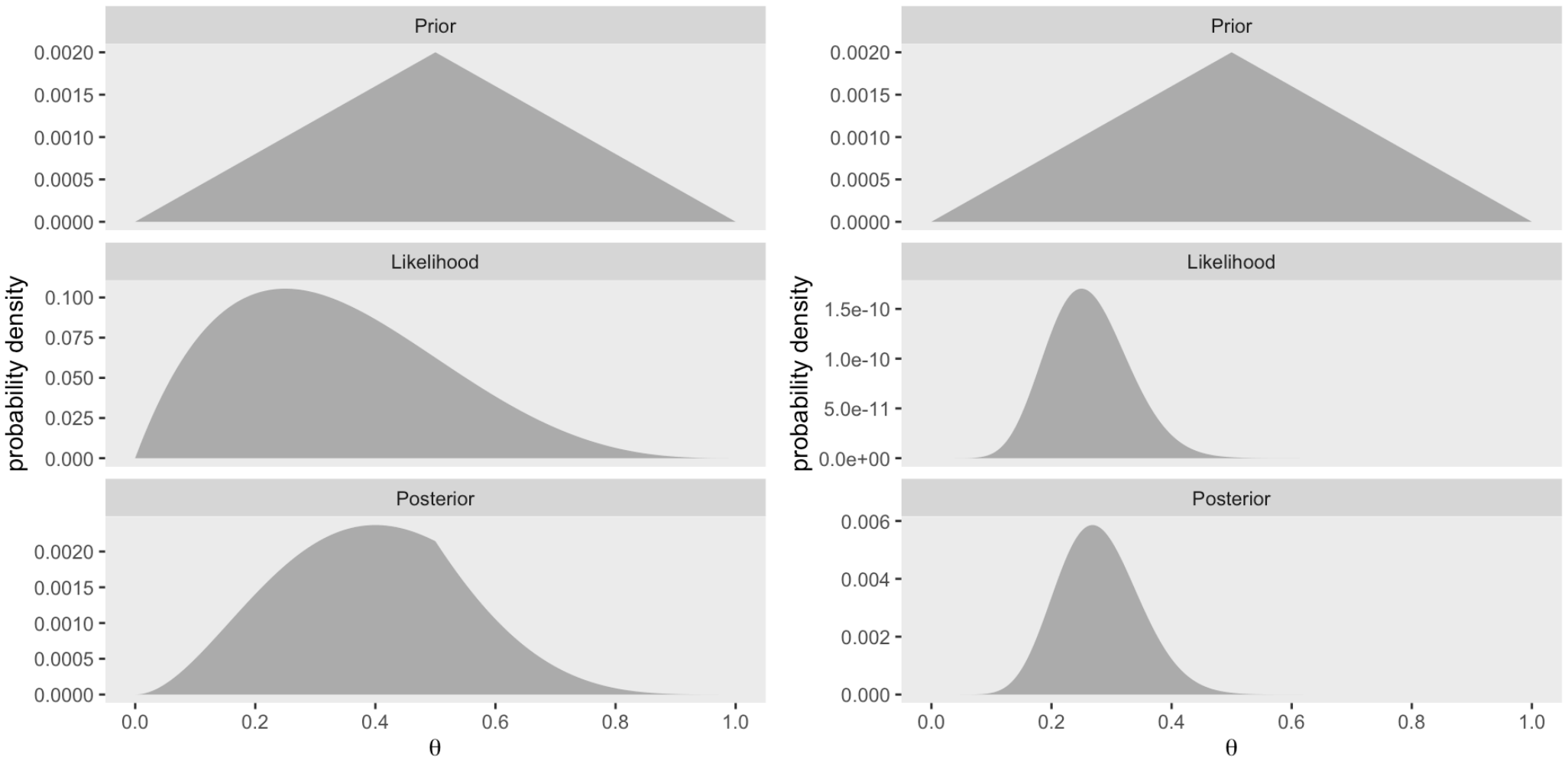
p(data| θ) is the likelihood.

Combining the three components

Priors influencing Posterior



sample size influence



Going from prior to posterior

What is our regression estimate again?

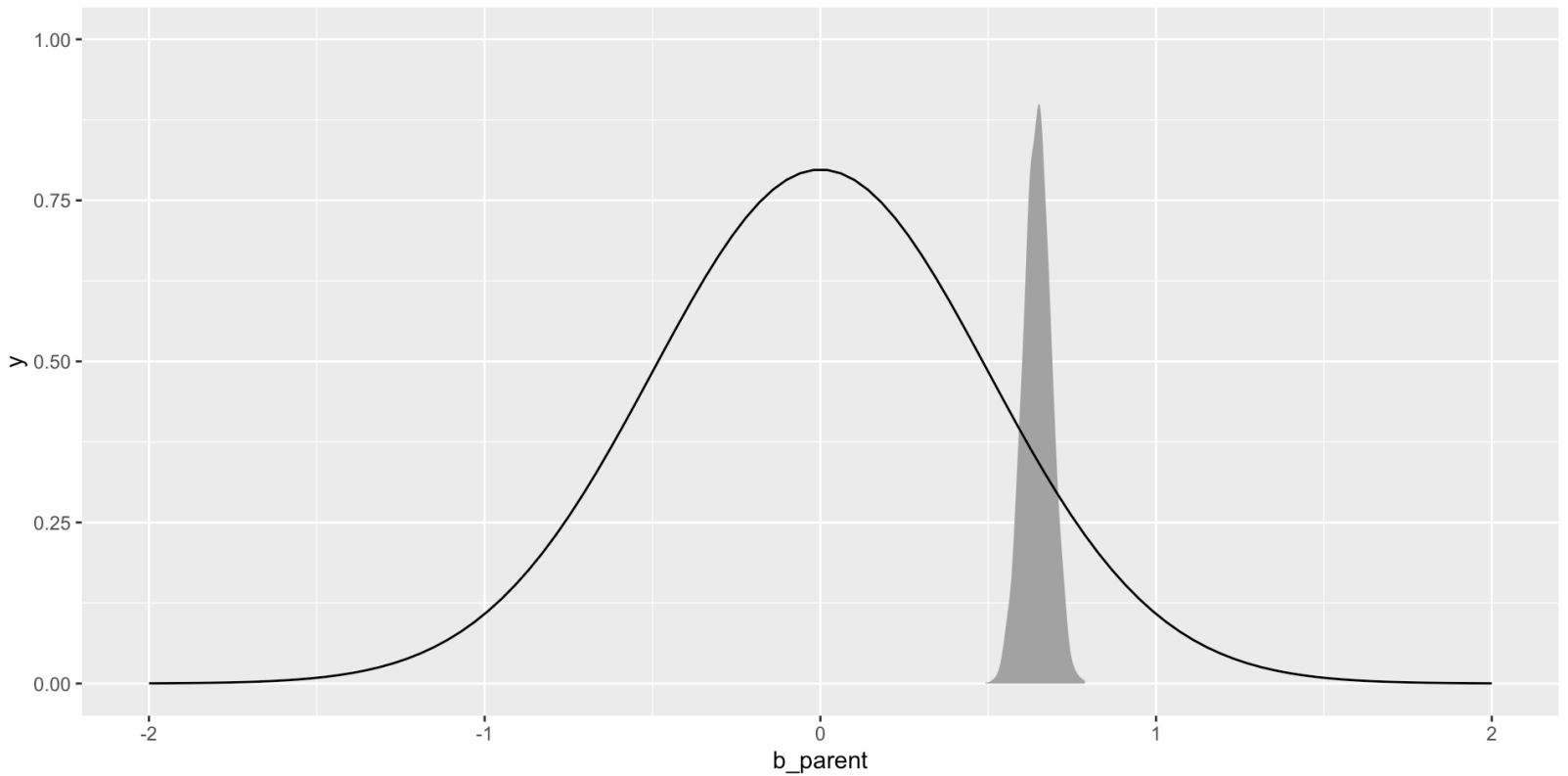
▶ Code

```
# A tibble: 1 × 6
  b_parent .lower .upper .width .point .interval
     <dbl>  <dbl>  <dbl>  <dbl> <chr>  <chr>
1    0.651  0.567  0.730   0.95 mode   hdi
```

Going from prior to posterior

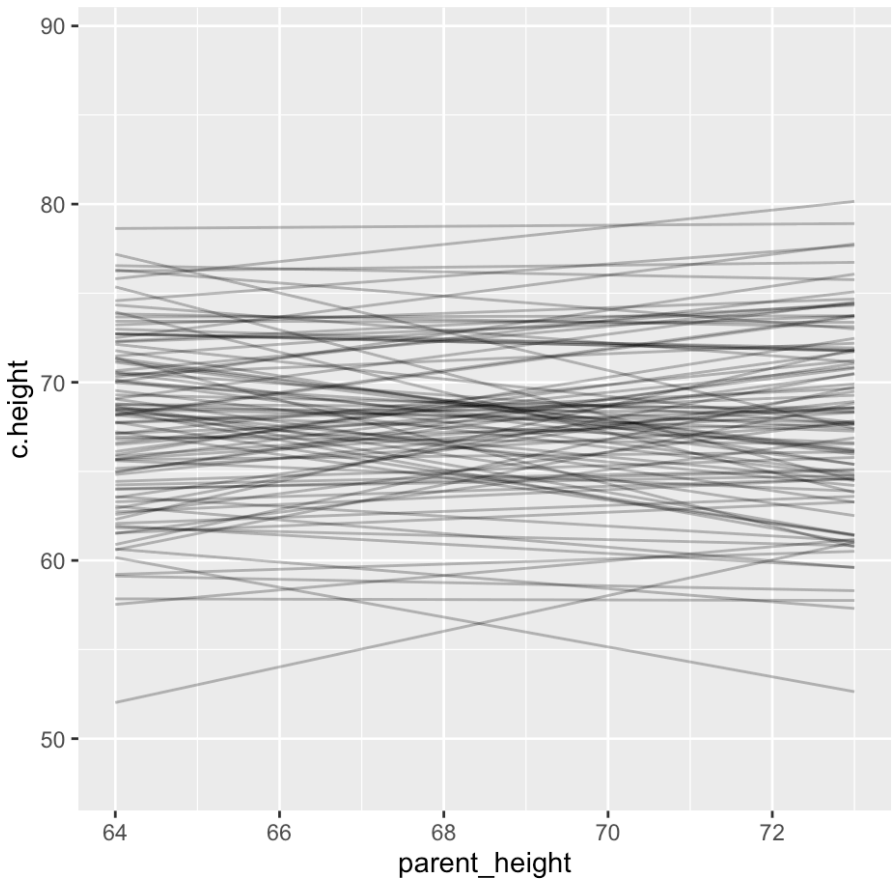
With a prior for b0 of N(0, .5)

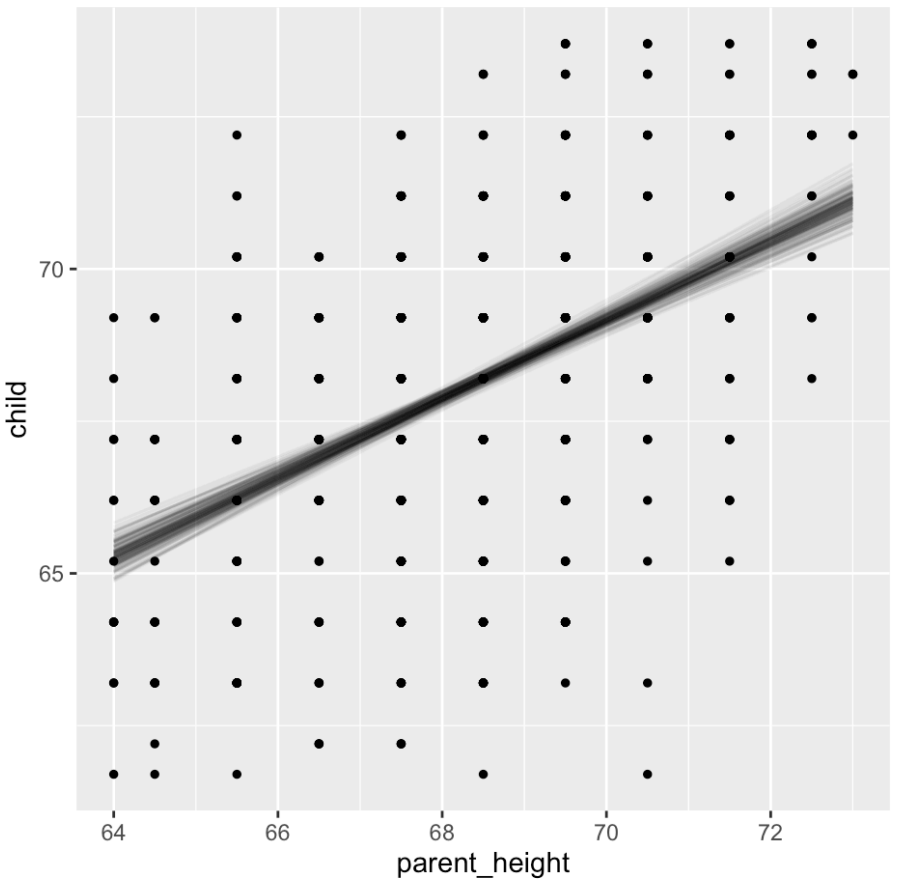
▶ Code



Going from prior to posterior

- plusible lines prior to data –> plausible lines after data





What is confusing:

Learn more - take my Bayes Class!

- Basically an advanced glm class, bayes is sort of extra.
- Easy graphing, CIs, model evaluation.
- Create generative model, where you can simulate data more directly.