Lab Exercise - STAN/MCMC

Luis Correia - Student No. 1006508566

February 12th 2020

Introduction

Today we will be starting off using Stan, looking at the kid's test score data set (available in resources for the Gelman Hill textbook).

```r
library(tidyverse)
library(rstan)
library(tidybayes)
library(ggplot2)
library(GGally)
```

The data look like this:

```r
kidiq <- read_rds("kidiq.RDS")
kidiq <- kidiq %>%
  mutate(cod_mhs = ifelse(mom_hs==0,"No-HS","HS"))
```

As well as the kid's test scores, we have a binary variable indicating whether or not the mother completed high school, the mother's IQ and age.

Descriptives

Question 1

Use plots or tables to show three interesting observations about the data. Remember:

- Explain what your graph/ tables show
- Choose a graph type that's appropriate to the data type

```r
kidiq %>%
  ggpairs(columns = c(1,3:4), ggplot2::aes(colour=cod_mhs))+
  theme_bw()
```

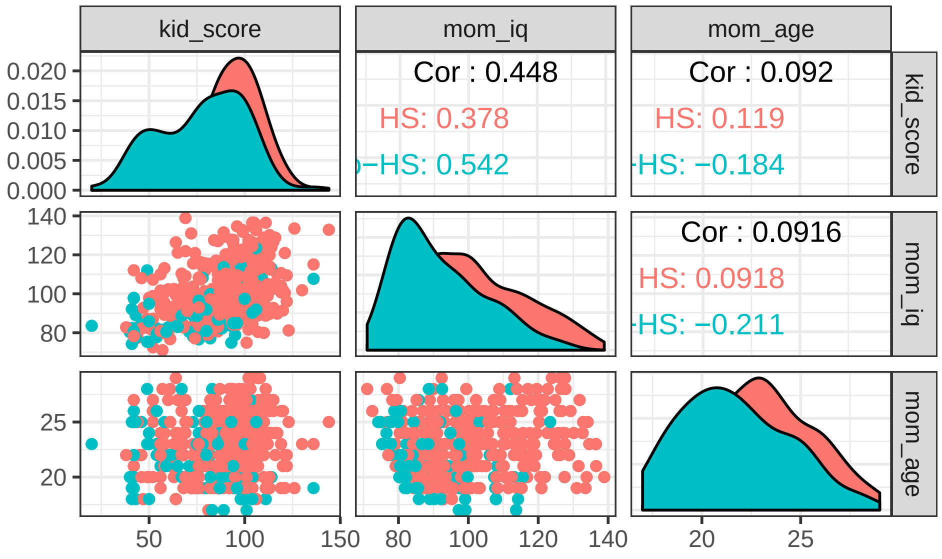


Figure 1: Kid Score vs Mom's I.Q.

This graph shows the densities variables for each group (HS and Non-HS), scattetplots of pairs of variables, showing its correlation.

From Graphs 1-3, we can summarize some behaviours observed, as follows:

- The variable `kid_score` is concentrated around 100 for group 'HS' and apparently having a bi-modal behaviour for group 'Non-HS', splitting between 50 and 95, with more concentration on the second peak. A positive correlation with `mom_iq` might suggest high I.Q. influences positively in kid's score;
- `mom_iq` is distributed differently for mom's high-school groups, with concentration around 85-90 for 'Non-HS' moms and 100 for 'HS' group. We noted possibly greater variability for 'HS' group as well. Correlation between 'age' and 'mom I.Q.' has opposite behaviour between groups, with positive correlation on group 'HS' and negative for group 'Non-HS'. This may suggest that greater `mom_age` are associated in opposite way with `mom_iq` variable;
- `mom_age` distributions shows slight different distribution with group 'Non-HS' concentrated around 20-22 and around 24 for 'HS' group. In the same way as, the correlation between 'age' and 'kid_score' has opposite behaviour between groups, suggesting the greater `mom_age` influences in opposite ways the `kid_score` variable: positivelly to 'HS' group and negativelly to 'Non-HS' group.

Estimating mean, no covariates

In class we were trying to estimate the mean and standard deviation of the kid's test scores. The `kids2.stan` file contains a Stan model to do this. If you look at it, you will notice the first `data` chunk lists some inputs that we have to define: the outcome variable `y`, number of observations `N`, and the mean and standard deviation of the prior on `mu`. Let's define all these values in a `data` list.

```
y <- kidiq$kid_score
mu0 <- 80
sigma0 <- 100

data <- list(y = y,
             N = length(y),
             mu0 = mu0,
             sigma0 = sigma0)
```

Now we can run the model:

```
fit <- stan(file = "kids2.stan",
            data = data)
```

```
##
## SAMPLING FOR MODEL 'kids2' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.031 seconds (Warm-up)
## Chain 1:                0.028 seconds (Sampling)
## Chain 1:                0.059 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'kids2' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.051 seconds (Warm-up)
## Chain 2:                0.043 seconds (Sampling)
## Chain 2:                0.094 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'kids2' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.047 seconds (Warm-up)
## Chain 3:                0.026 seconds (Sampling)
## Chain 3:                0.073 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'kids2' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.035 seconds (Warm-up)
## Chain 4:                0.03 seconds (Sampling)
## Chain 4:                0.065 seconds (Total)
## Chain 4:
```

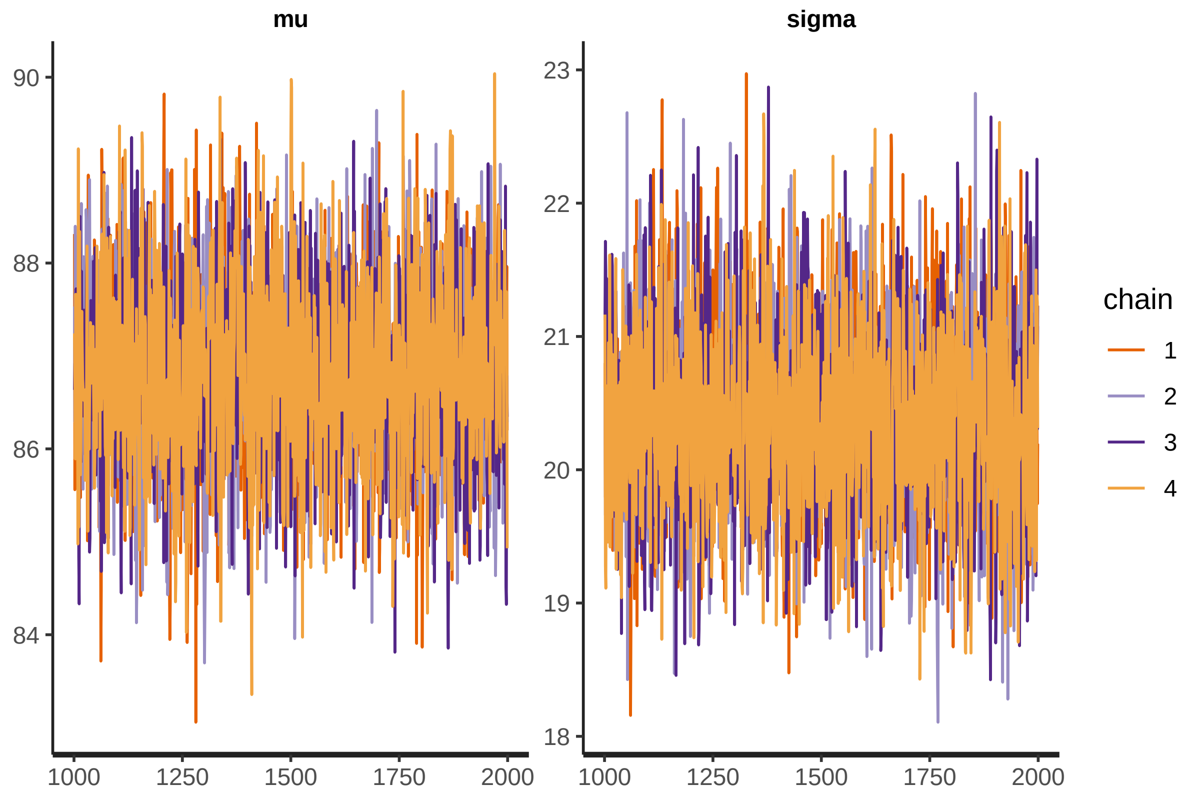
Look at the summary

```
fit
```

```
## Inference for Stan model: kids2.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##            mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff
## mu        86.81    0.02 0.97    84.89    86.19    86.78    87.45    88.70  3621
## sigma     20.38    0.01 0.70    19.06    19.90    20.37    20.83    21.80  3665
## lp__   -1525.54    0.02 1.02 -1528.20 -1525.91 -1525.24 -1524.82 -1524.56  1995
##        Rhat
## mu        1
## sigma     1
## lp__      1
##
## Samples were drawn using NUTS(diag_e) at Thu Feb 13 16:03:15 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Traceplot

```
traceplot(fit)
```



All looks fine.

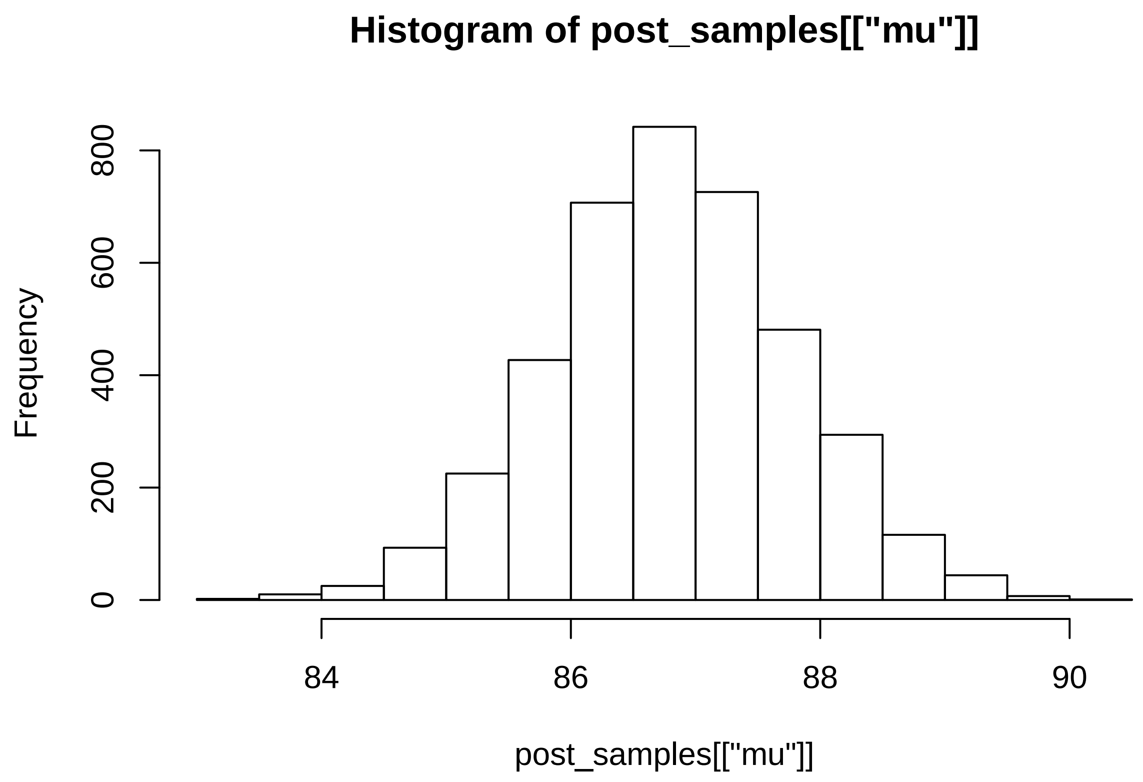
Understanding output

What does the model actually give us? A number of samples from the posteriors. To see this, we can use `extract` to get the samples.

```
post_samples <- extract(fit)
```

This is a list, and in this case, each element of the list has 4000 samples. E.g. quickly plot a histogram of mu

```
hist(post_samples[["mu"]])
```



Plot estimates

There are a bunch of packages, built-in functions that let you plot the estimates from the model, and I encourage you to explore these options (particularly in `bayesplot`, which we will most likely be using later on). I like using the `tidybayes` package, which allows us to easily get the posterior samples in a tidy format (e.g. using gather draws to get in long format). Once we have that, it's easy to just pipe and do ggplots as usual. `tidybayes` also has a bunch of fun visualizations, see more info here: https://mjskay.github.io/tidybayes/articles/tidybayes.html#introduction

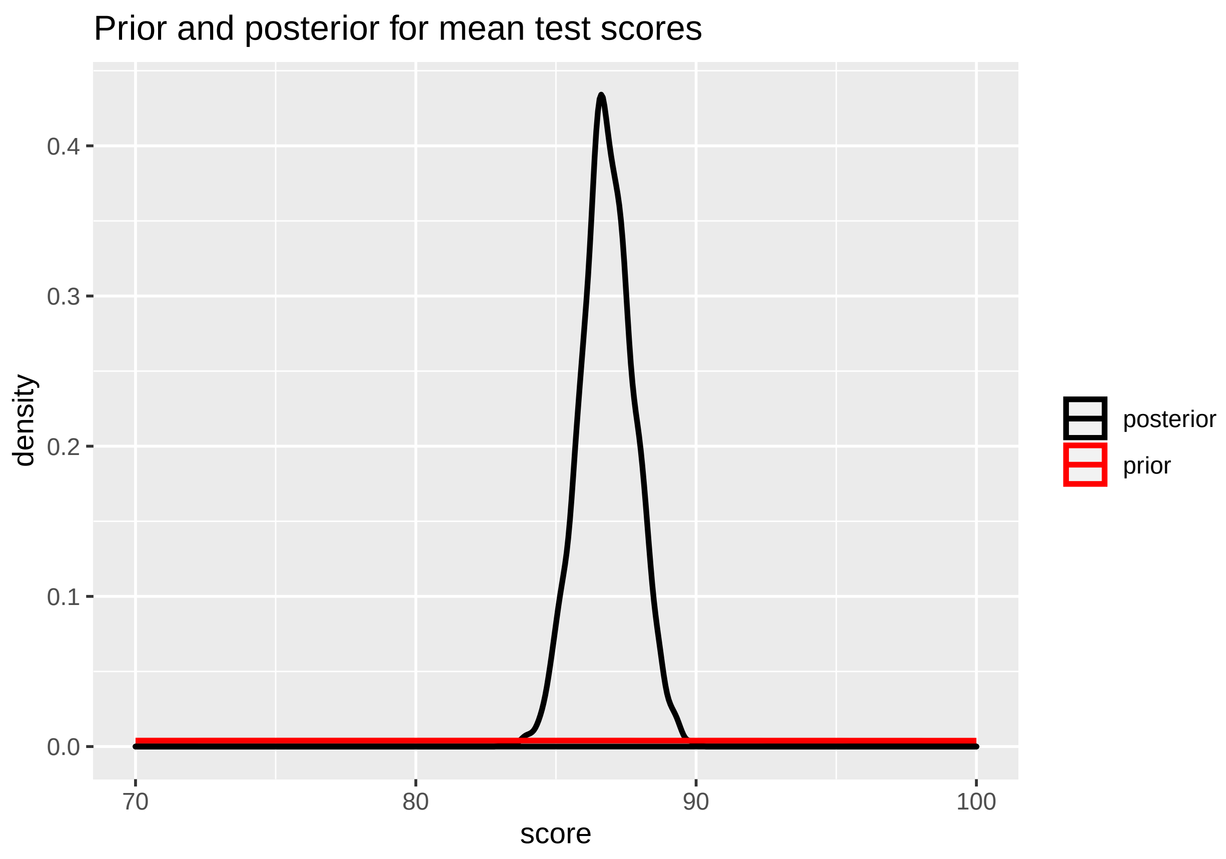
Get the posterior samples for mu and sigma in long format:

```
dsamples <- fit %>%
  gather_draws(mu, sigma)
dsamples
```

```
## # A tibble: 8,000 x 5
## # Groups:   .variable [2]
##    .chain .iteration .draw .variable .value
##      <int>      <int> <int> <chr>      <dbl>
##   1      1          1     1 mu          88.3
##   2      1          2     2 mu          85.6
##   3      1          3     3 mu          88.1
##   4      1          4     4 mu          86.1
##   5      1          5     5 mu          87.2
##   6      1          6     6 mu          87.2
##   7      1          7     7 mu          86.5
##   8      1          8     8 mu          86.4
##   9      1          9     9 mu          87.4
## 10       1         10    10 mu          85.8
## # ... with 7,990 more rows
```

Let's plot the density of the posterior samples for mu and add in the prior distribution

```
dsamples %>%
  filter(.variable == "mu") %>%
  ggplot(aes(.value, color = "posterior")) + geom_density(size = 1) +
  xlim(c(70, 100)) +
  stat_function(fun = dnorm,
        args = list(mean = mu0,
                    sd = sigma0),
        aes(colour = 'prior'), size = 1) +
  scale_color_manual(name = "", values = c("prior" = "red", "posterior" = "black")) +
  ggtitle("Prior and posterior for mean test scores") +
  xlab("score")
```



Question 2

Change the prior to be much more informative (by changing the standard deviation to be 0.1). Rerun the model. Do the estimates change? Plot the prior and posterior densities.

```
fitnew <- stan(file = "kids2a.stan",
            data = data)
```

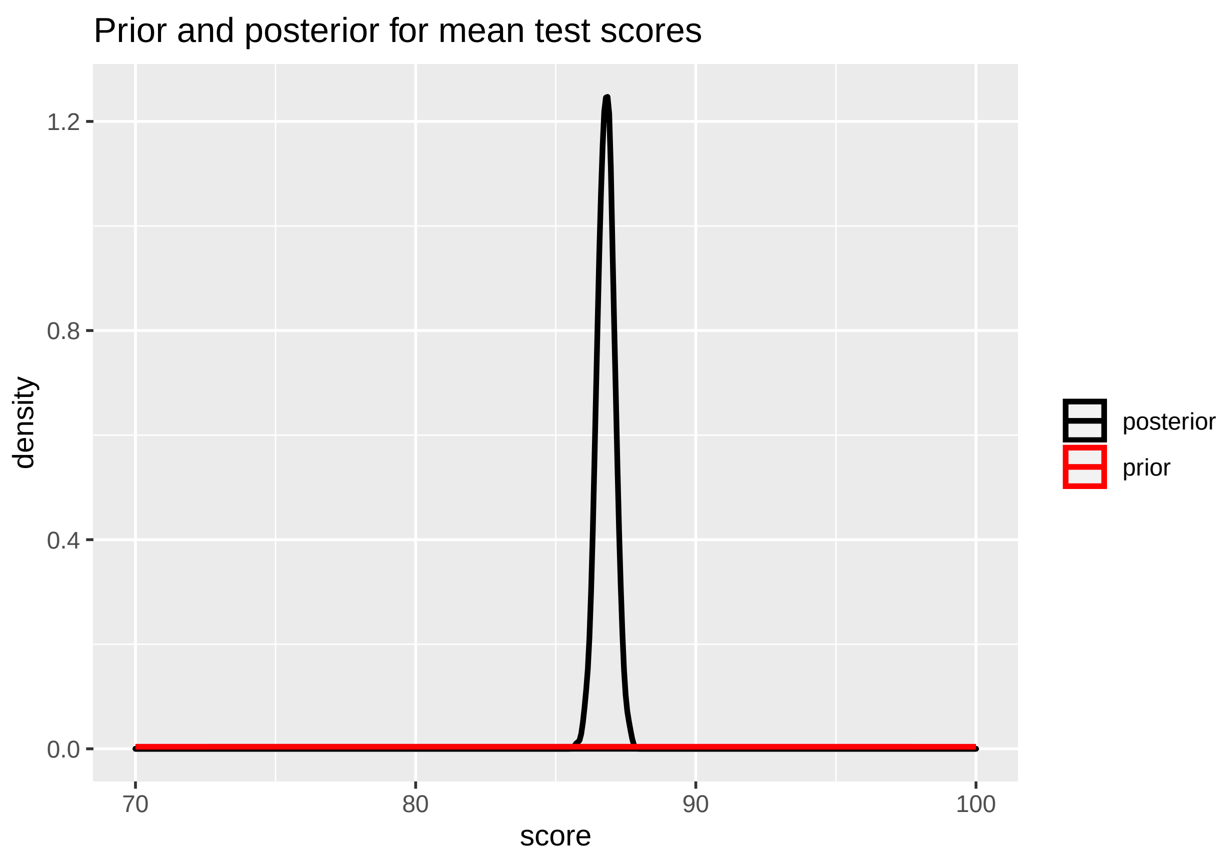
```
##
## SAMPLING FOR MODEL 'kids2a' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0.001 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.031 seconds (Warm-up)
## Chain 1:                0.022 seconds (Sampling)
## Chain 1:                0.053 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'kids2a' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.064 seconds (Warm-up)
## Chain 2:                0.026 seconds (Sampling)
## Chain 2:                0.09 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'kids2a' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.027 seconds (Warm-up)
## Chain 3:                0.021 seconds (Sampling)
## Chain 3:                0.048 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'kids2a' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.039 seconds (Warm-up)
## Chain 4:                0.022 seconds (Sampling)
## Chain 4:                0.061 seconds (Total)
## Chain 4:
```

```
fitnew
```

```
## Inference for Stan model: kids2a.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##            mean se_mean   sd     2.5%      25%      50%      75%    97.5% n_eff
## mu        86.79    0.01 0.31    86.17    86.58    86.79    87.00    87.40  3341
## sigma      6.35    0.00 0.05     6.26     6.32     6.35     6.39     6.45  4156
## lp__   -5054.34    0.02 1.01 -5056.98 -5054.77 -5054.06 -5053.60 -5053.33  2037
##        Rhat
## mu        1
## sigma     1
## lp__      1
##
## Samples were drawn using NUTS(diag_e) at Thu Feb 13 16:04:31 2020.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

The estimates of Sigma decreased from aroubd 20.6 to 6.5.

```
dsamplesnew <- fitnew %>%
  gather_draws(mu, sigma)
dsamplesnew %>%
  filter(.variable == "mu") %>%
  ggplot(aes(.value, color = "posterior")) + geom_density(size = 1) +
  xlim(c(70, 100)) +
  stat_function(fun = dnorm,
        args = list(mean = mu0,
                    sd = sigma0),
        aes(colour = 'prior'), size = 1) +
  scale_color_manual(name = "", values = c("prior" = "red", "posterior" = "black")) +
  ggtitle("Prior and posterior for mean test scores") +
  xlab("score")
```



Adding covariates

Now let's see how kid's test scores are related to mother's education. We want to run the simple linear regression

$$Score = \alpha + \beta X$$

where $X = 1$ if the mother finished high school and zero otherwise.

`kid3.stan` has the stan model to do this. Notice now we have some inputs related to the design matrix X and the number of covariates (in this case, it's just 1).

Let's get the data we need and run the model.

```
X <- as.matrix(kidiq$mom_hs, ncol = 1)
K <- 1

data <- list(y = y, N = length(y),
             X =X, K = K)
fit2 <- stan(file = "kids3.stan",
            data = data,
            iter = 1000)
```

```
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 1: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 1: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 1: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 1: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 1: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 1: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 1: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 1: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 1: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 1: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 1: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.237 seconds (Warm-up)
## Chain 1:                0.21 seconds (Sampling)
## Chain 1:                0.447 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 2: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 2: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 2: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 2: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 2: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 2: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 2: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 2: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 2: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 2: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 2: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.336 seconds (Warm-up)
## Chain 2:                0.231 seconds (Sampling)
## Chain 2:                0.567 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
```

```
## Chain 3: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 3: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 3: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 3: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 3: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 3: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 3: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 3: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 3: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 3: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 3: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 3: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.375 seconds (Warm-up)
## Chain 3:                0.196 seconds (Sampling)
## Chain 3:                0.571 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 4: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 4: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 4: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 4: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 4: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 4: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 4: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 4: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 4: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 4: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 4: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.266 seconds (Warm-up)
## Chain 4:                0.163 seconds (Sampling)
## Chain 4:                0.429 seconds (Total)
## Chain 4:
```

Question 3

Confirm that the estimates of the intercept and slope are comparable to results from `lm()`

```r
fitlm <- lm(kid_score ~ mom_hs, data = kidiq)
summary(fitlm)
```

```
##
## Call:
## lm(formula = kid_score ~ mom_hs, data = kidiq)
##
## Residuals:
```

```
##    Min     1Q Median     3Q    Max
## -57.55 -13.32   2.68  14.68  58.45
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   77.548      2.059  37.670  < 2e-16 ***
## mom_hs        11.771      2.322   5.069 5.96e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.85 on 432 degrees of freedom
## Multiple R-squared:  0.05613,    Adjusted R-squared:  0.05394
## F-statistic: 25.69 on 1 and 432 DF,  p-value: 5.957e-07
```

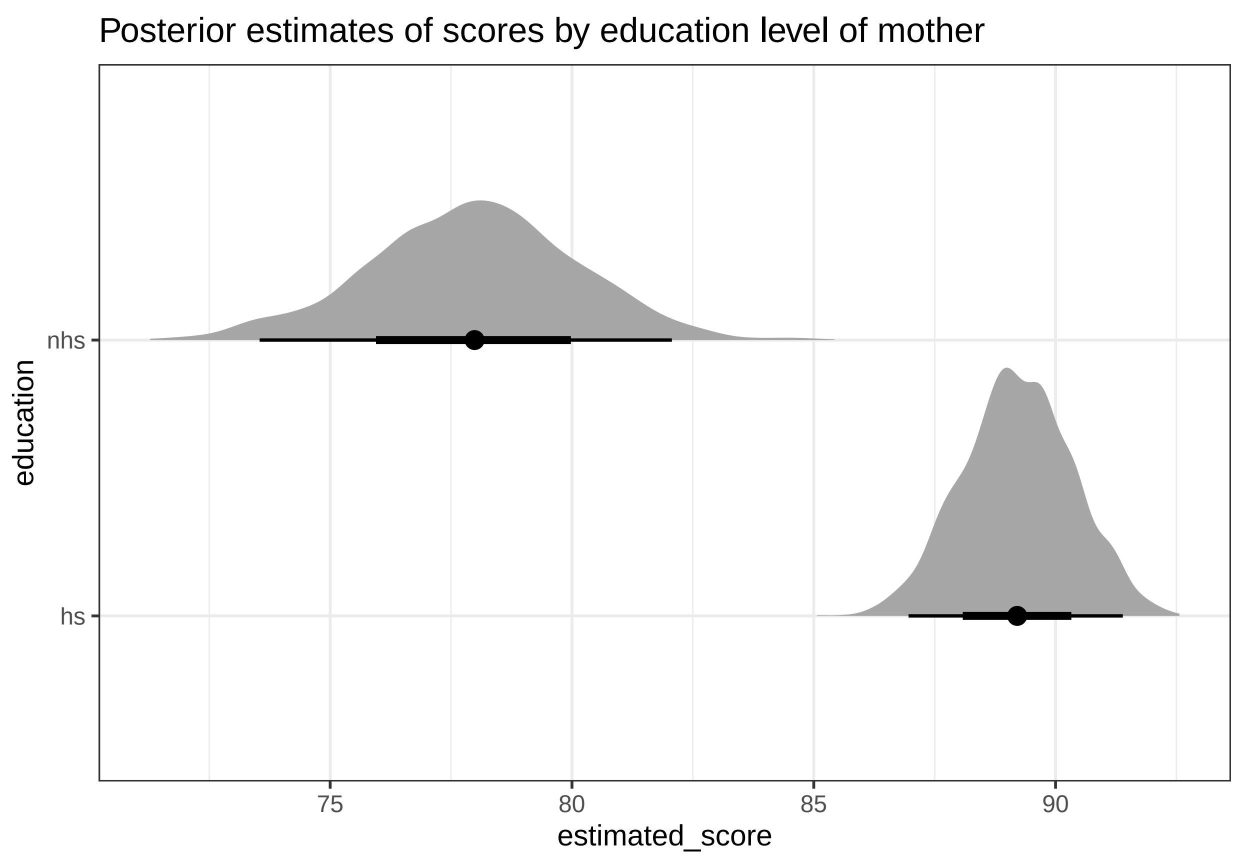
```r
summary(fit2)[["summary"]][1:2,]
```

```
##             mean   se_mean       sd      2.5%       25%      50%      75%
## alpha   77.95572 0.0926319 2.140531 73.540704 76.563638 77.98631 79.33377
## beta[1] 11.26147 0.1043563 2.388776  6.751225  9.688243 11.17231 12.84998
##            97.5%    n_eff     Rhat
## alpha   82.06656 533.9761 1.010584
## beta[1] 16.20029 523.9780 1.012126
```

Plotting results

It might be nice to plot the posterior samples of the estimates for the non-high-school and high-school mothered kids. Here's some code that does this: notice the `beta[condition]` syntax. Also notice I'm using `spread_draws`, because it's easier to calculate the estimated effects in wide format

```r
fit2 %>%
  spread_draws(alpha, beta[condition], sigma) %>%
     mutate(nhs = alpha, # no high school is just the intercept
          hs = alpha + beta) %>%
  pivot_longer(nhs:hs, names_to = "education", values_to = "estimated_score") %>%
  ggplot(aes(y = education, x = estimated_score)) +
  stat_halfeyeh() +
  theme_bw() +
  ggtitle("Posterior estimates of scores by education level of mother")
```



Question 4

Add in mother's IQ as a covariate and rerun the model. You will probably want to mean center the covariate before putting it into the model. Interpret the coefficient on the (centered) mum's IQ.

Including Mother's IQ the simple linear regression becomes like this:

$$Score = \alpha + \beta X + \beta_2 X_2$$

where X_2 is mother's I.Q. and the stan model is on `kid4.stan` file.

```r
X2 <- cbind(X, as.matrix(kidiq$mom_iq, ncol = 1))
K <- 2

data <- list(y = y, N = length(y),
             X = X2, K = K)
fit3 <- stan(file = "kids3.stan",
            data = data,
            iter = 1000)
```

```
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 0 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
```

```
## Chain 1:
## Chain 1: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 1: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 1: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 1: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 1: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 1: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 1: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 1: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 1: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 1: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 1: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 1: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 1.335 seconds (Warm-up)
## Chain 1:                0.943 seconds (Sampling)
## Chain 1:                2.278 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 0.001 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 10 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 2: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 2: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 2: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 2: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 2: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 2: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 2: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 2: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 2: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 2: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 2: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 2.126 seconds (Warm-up)
## Chain 2:                0.766 seconds (Sampling)
## Chain 2:                2.892 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 0 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 3: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 3: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 3: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 3: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 3: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 3: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 3: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 3: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 3: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 3: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 3: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 1.821 seconds (Warm-up)
## Chain 3:                0.582 seconds (Sampling)
## Chain 3:                2.403 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'kids3' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 0 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 4: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 4: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 4: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 4: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 4: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 4: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 4: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 4: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 4: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 4: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 4: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 1.522 seconds (Warm-up)
## Chain 4:                0.701 seconds (Sampling)
## Chain 4:                2.223 seconds (Total)
## Chain 4:
```

Question 5

Confirm the results from Stan agree with `lm()`

```r
fitlm3 <- lm(kid_score ~ mom_hs + mom_iq, data = kidiq)
summary(fitlm3)
```

```
##
## Call:
## lm(formula = kid_score ~ mom_hs + mom_iq, data = kidiq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.873 -12.663   2.404  11.356  49.545
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 25.73154    5.87521   4.380 1.49e-05 ***
## mom_hs       5.95012    2.21181   2.690  0.00742 **
## mom_iq       0.56391    0.06057   9.309  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.14 on 431 degrees of freedom
## Multiple R-squared:  0.2141, Adjusted R-squared:  0.2105
## F-statistic: 58.72 on 2 and 431 DF,  p-value: < 2.2e-16
```

```
summary(fit3)[["summary"]][1:3,]
```

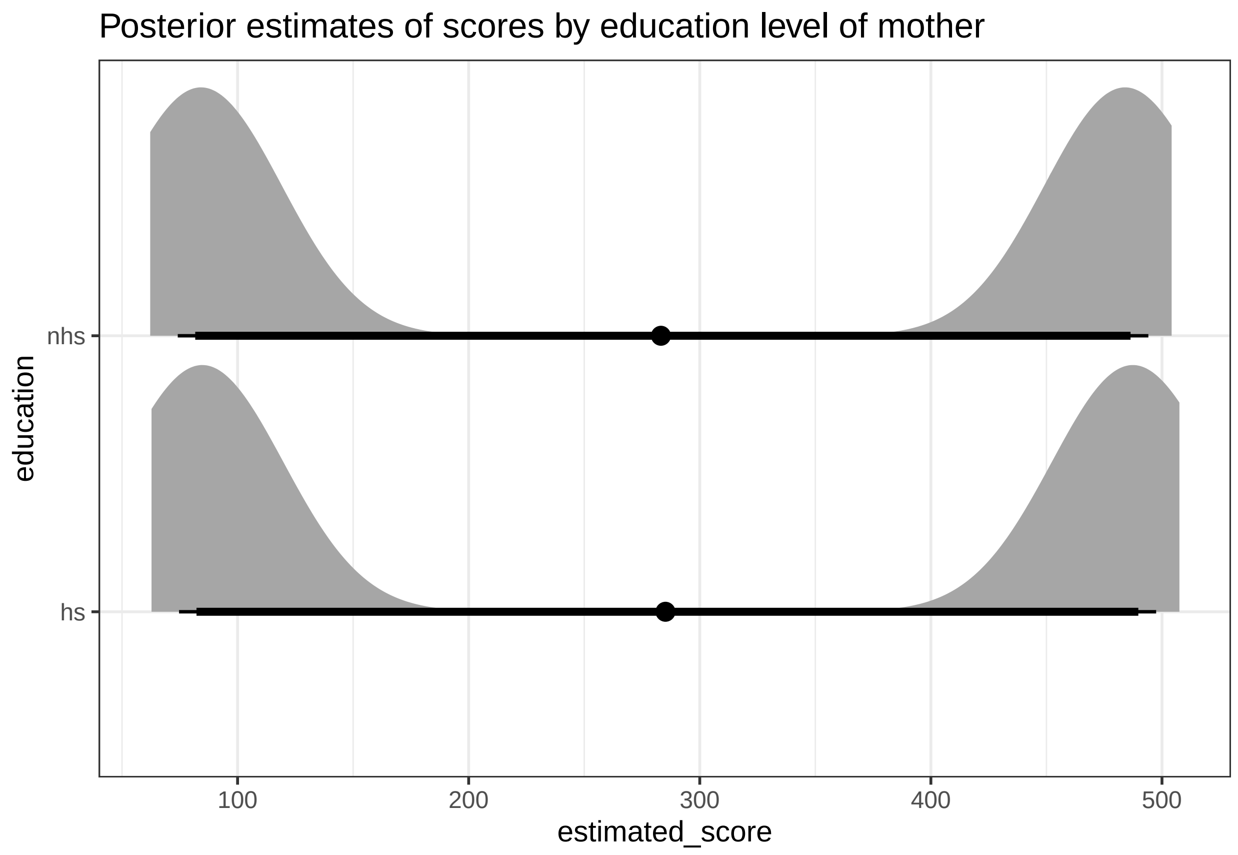
```
##               mean     se_mean         sd       2.5%        25%        50%
## alpha   25.9218839 0.222188422 6.14367496 13.5667422 21.7876376 26.0963512
## beta[1]  5.6427010 0.068983002 2.21668354  1.4424484  4.0682120  5.6514783
## beta[2]  0.5645238 0.002309063 0.06317139  0.4421151  0.5220528  0.5646071
##               75%      97.5%     n_eff     Rhat
## alpha   29.997695 37.8822562  764.5636 1.003747
## beta[1]  7.168641  9.9869614 1032.5788 1.003197
## beta[2]  0.604275  0.6903281  748.4614 1.005002
```

Question 6

Plot the posterior estimates of scores by education of mother for mothers who have an IQ of 110.

@Monica - I'm not sure about how to gather the coefficents on model/draws (I'm probably wrong). Could you please correct my code below if necessary? Thank you so much!

```
fit3 %>%
  spread_draws(alpha, beta[condition], sigma) %>%
     mutate(nhs = alpha + beta[2]*110,          # no high school is just the intercept
            hs = alpha + beta[1] + beta[2]*110) %>%
  pivot_longer(nhs:hs, names_to = "education", values_to = "estimated_score") %>%
  ggplot(aes(y = education, x = estimated_score)) +
  stat_halfeyeh() +
  theme_bw() +
  ggtitle("Posterior estimates of scores by education level of mother")
```


Posterior estimates of scores by education level of mother
education
nhs
hs
estimated_score
100
200
300
400
500