Introduction to Stan

Monica Alexander

February 9 2021

Contents

1 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).

```
library(tidyverse)
library(rstan)
library(tidybayes)
library(here)
library(corrplot)
```

The data look like this:

```
kidiq <- read_rds(here("data","kidiq.RDS"))
```

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.

2 Descriptives

2.1 Question 1

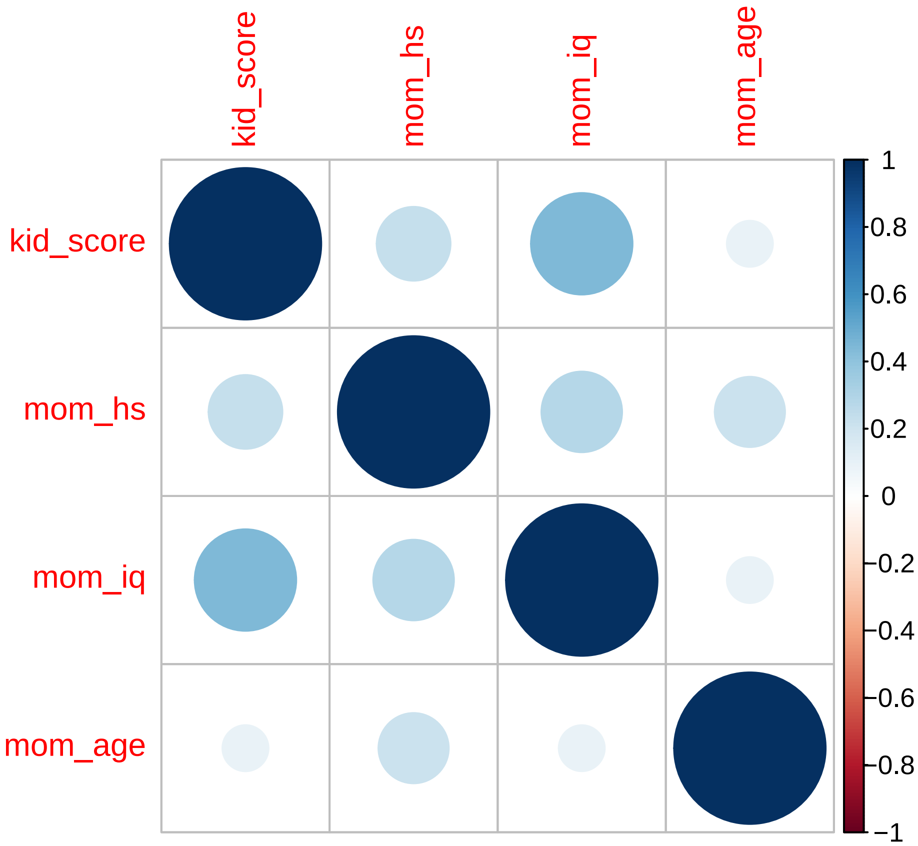
Use plots or tables to show three interesting observations about the data. Remember:

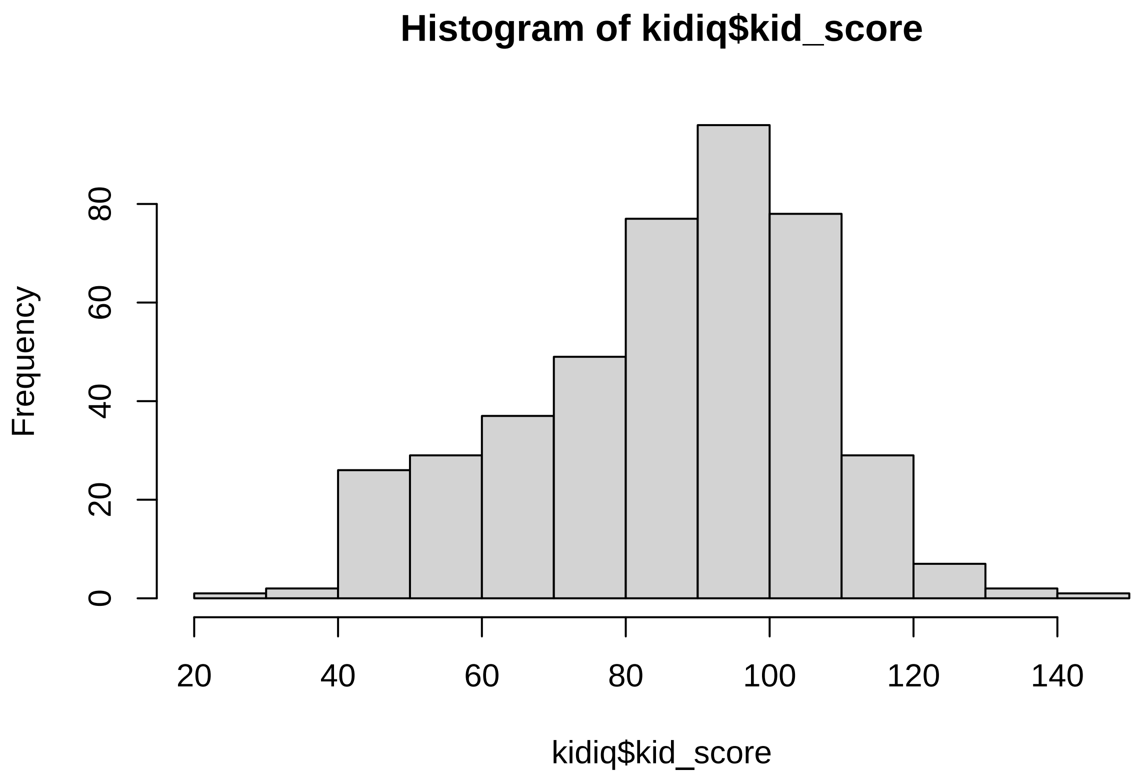
- Explain what your graph/ tables show

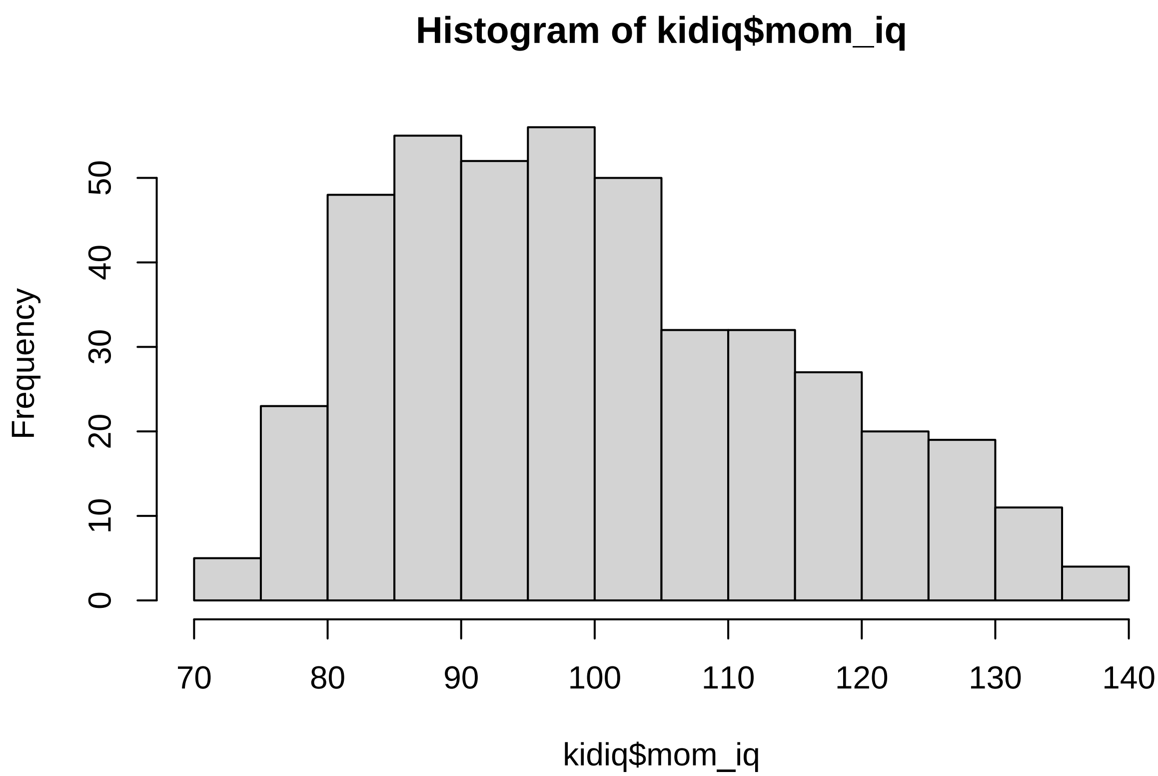
- Choose a graph type that's appropriate to the data type

Answer:

- We want to explore the correlation between the kid_score and the mom_IQ first. First we could find the correlation between kid_score and mom_ID are the highest among the other three factors.
- Then then further explore the distribution of Mom_IQ and kids_score. Even though the correlation is high. But we find they have different type of distributions. The kid_score skew to the left and the mom's IQ skew to the right. Which is an interesting phenomenon.







3 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 <- 10

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

Now we can run the model:

```
fit <- stan(file = "/Users/siyiwei/Desktop/applied-stats-2021/code/models/kids2.stan", data = data)
```

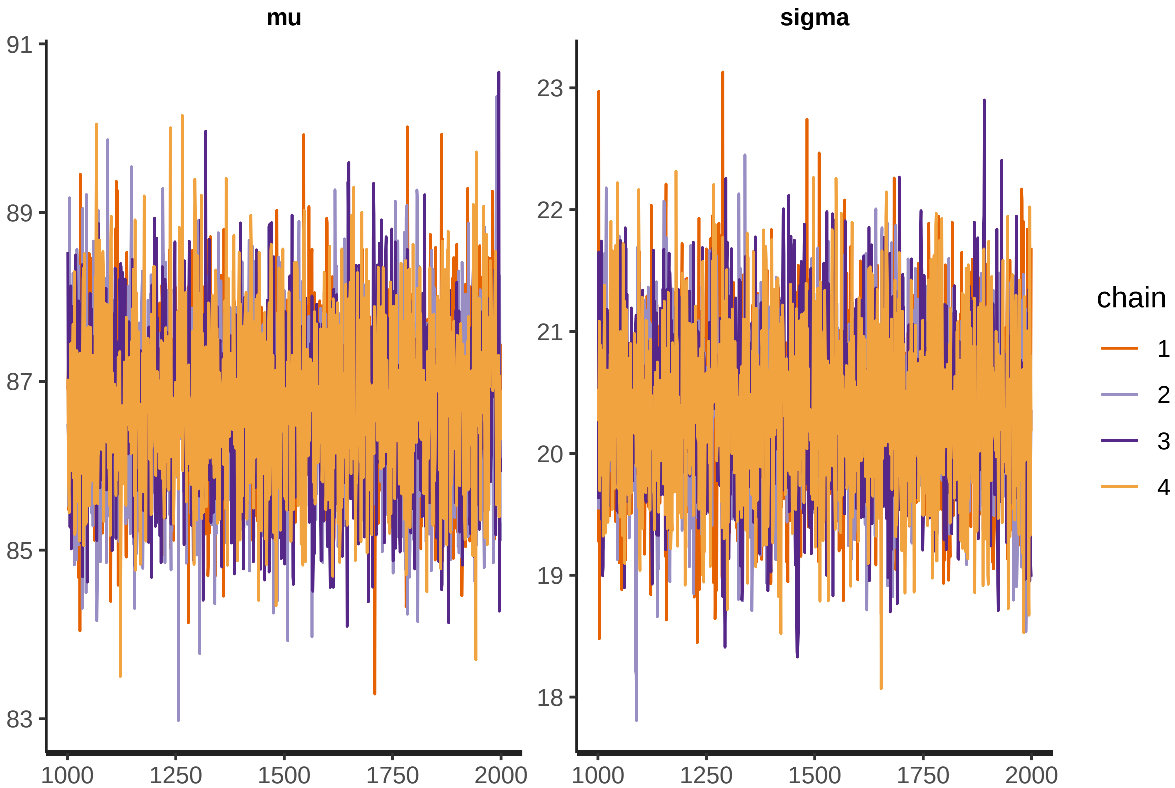
Look at the summary

```
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       86.75    0.02 0.97    84.86    86.11    86.75    87.37    88.67  3804
## sigma    20.37    0.01 0.68    19.05    19.90    20.36    20.83    21.71  3125
## lp__  -1525.75    0.02 1.01 -1528.39 -1526.12 -1525.44 -1525.04 -1524.79  1687
##       Rhat
## mu       1
## sigma    1
## lp__     1
##
## Samples were drawn using NUTS(diag_e) at Fri Feb 12 02:40:29 2021.
## 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)
```



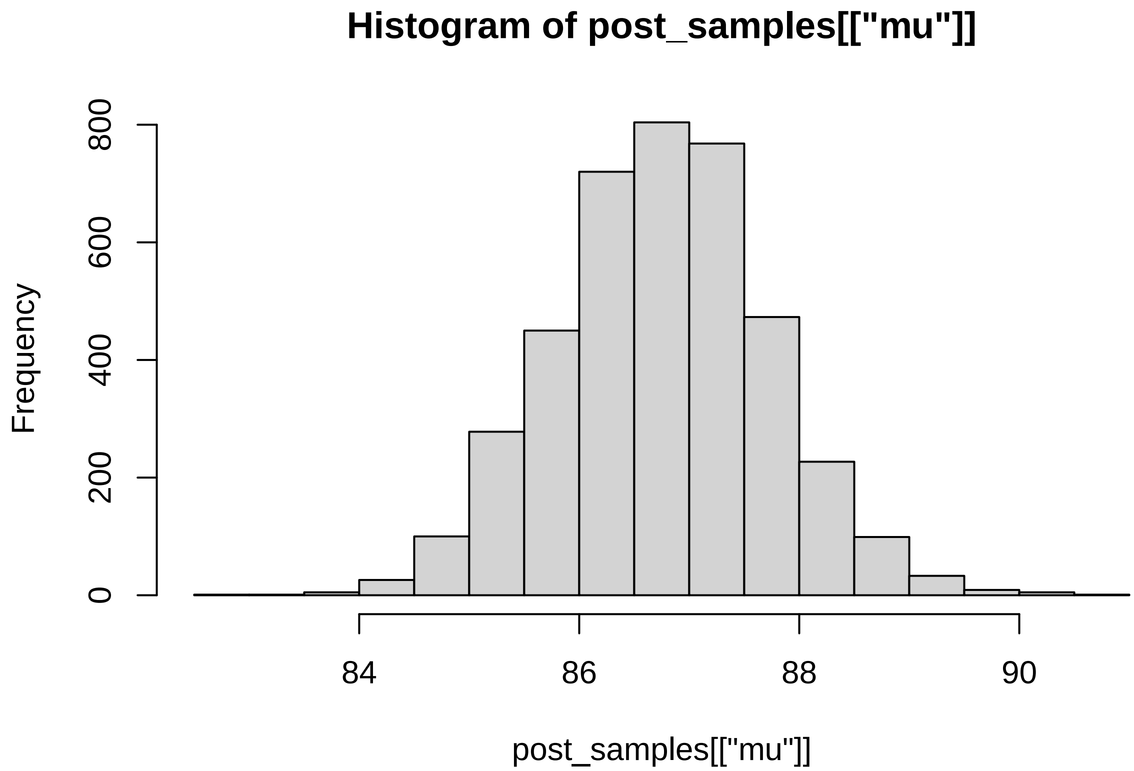
All looks fine.

3.1 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




```
## [1] 86.75325

##     2.5%
## 84.86419

##    97.5%
## 88.66543
```

3.2 Plot estimates

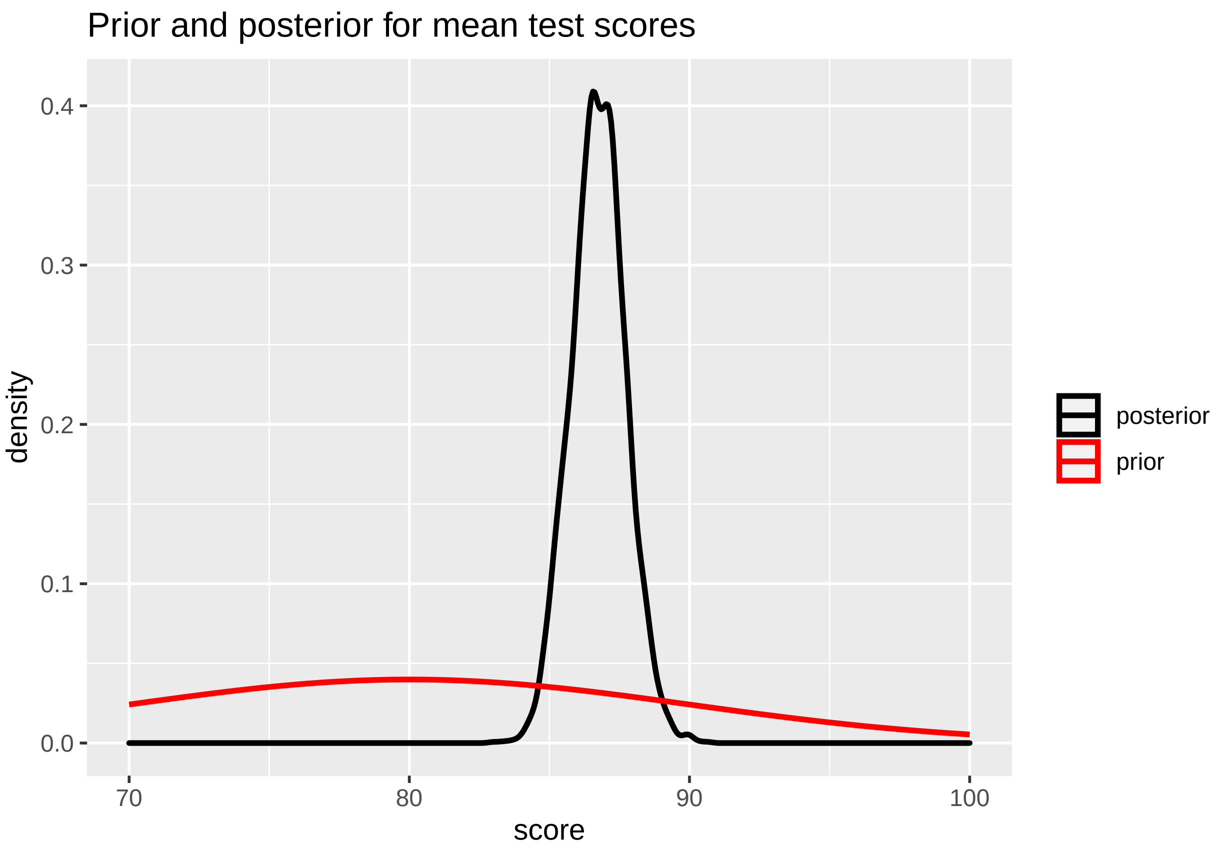
There are a bunch of packages, built-in functions that let you plot the estimates from the model, and I encourage you to explore these options (particularly in `bayesplot`, which we will most likely be using later on). I like using the `tidybayes` package, which allows us to easily get the posterior 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:

```
## # A tibble: 8,000 x 5
## # Groups:   .variable [2]
##    .chain .iteration .draw .variable .value
##     <int>      <int> <int> <chr>      <dbl>
##  1      1          1     1 mu          86.9
##  2      1          2     2 mu          87.2
##  3      1          3     3 mu          85.5
##  4      1          4     4 mu          86.0
##  5      1          5     5 mu          85.6
##  6      1          6     6 mu          86.5
##  7      1          7     7 mu          87.4
##  8      1          8     8 mu          87.3
##  9      1          9     9 mu          87.7
```

```
## 10      1         10    10 mu          85.9
## # ... with 7,990 more rows
```

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



3.3 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.

- The posterior estimates are getting much closer to 80. Moreover, the standard deviation of the posterior decreases a huge amount. Meaning we have a much more concentrated posterior distribution. The plots on the prior and posterior densities could verify this result.

```
y <- kidiq$kid_score
mu0 <- 80
sigma0 <- 0.1

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

fit <- stan(file = "/Users/siyiwei/Desktop/applied-stats-2021/code/models/kids2.stan", data = data)
```

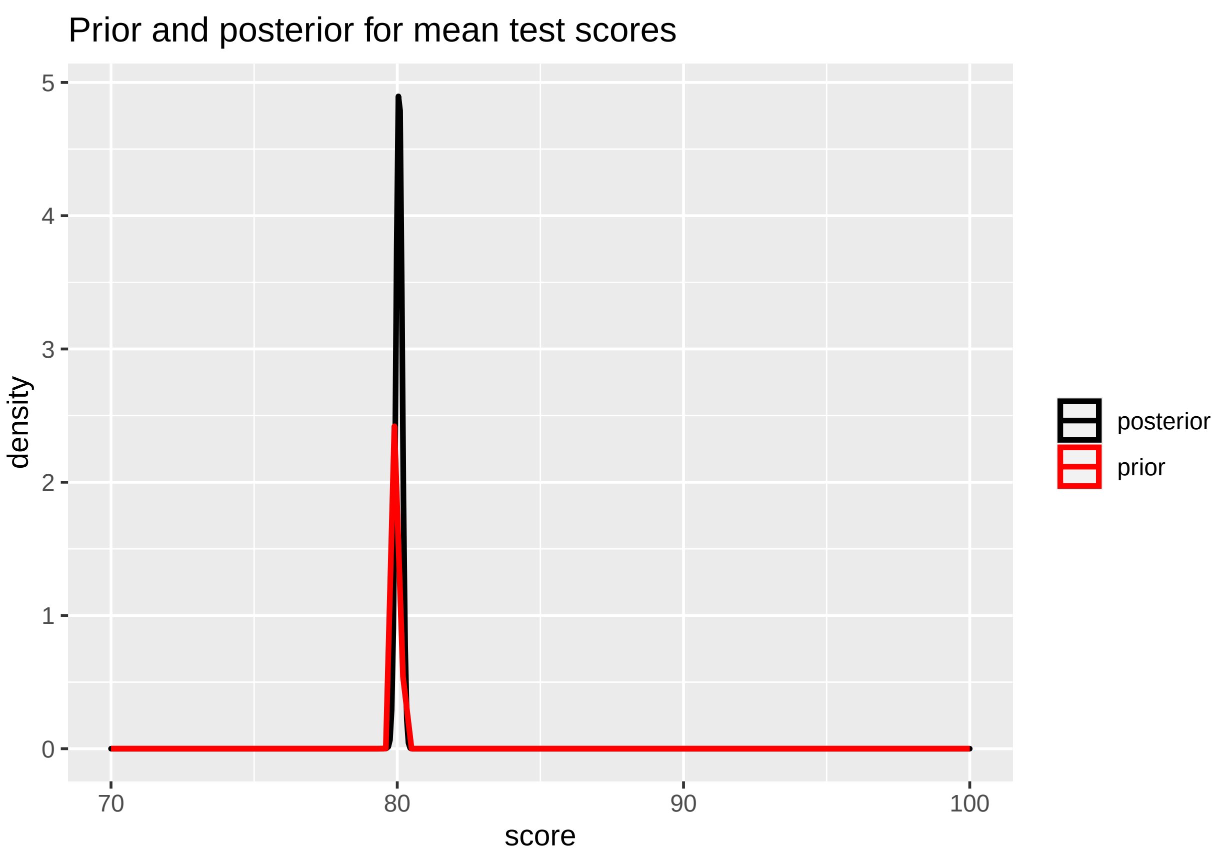
```
summary(fit)
```

```
## $summary
##              mean     se_mean        sd        2.5%          25%          50%
```

```
## mu       80.06312 0.001830445 0.1013631    79.86363    79.99319    80.06252
## sigma    21.42009 0.012308576 0.7499408    19.99378    20.89588    21.41008
## lp__  -1548.42549 0.025271343 1.0315983 -1551.17414 -1548.84952 -1548.10431
##              75%       97.5%    n_eff     Rhat
## mu       80.13140    80.25991 3066.520 1.000112
## sigma    21.92163    22.93823 3712.260 1.000378
## lp__  -1547.68286 -1547.39142 1666.344 1.000350
##
## $c_summary
## , , chains = chain:1
##
##          stats
## parameter        mean        sd        2.5%         25%         50%         75%
##     mu       80.06199 0.1037870    79.86719    79.98692    80.06131    80.13254
##     sigma    21.39985 0.7727731    19.98609    20.83457    21.36176    21.91662
##     lp__  -1548.48130 1.0326286 -1551.12813 -1548.92518 -1548.17947 -1547.73082
##          stats
## parameter       97.5%
##     mu       80.25647
##     sigma    23.03994
##     lp__  -1547.40046
##
## , , chains = chain:2
##
##          stats
## parameter        mean         sd        2.5%         25%         50%
##     mu       80.06876 0.09943907    79.87415    80.00128    80.06790
##     sigma    21.41486 0.76327030    19.95102    20.88687    21.41595
##     lp__  -1548.42896 1.06731411 -1551.43512 -1548.80994 -1548.09521
##          stats
## parameter         75%       97.5%
##     mu       80.13503    80.28454
##     sigma    21.92736    22.92152
##     lp__  -1547.67148 -1547.39031
##
## , , chains = chain:3
##
##          stats
## parameter        mean        sd        2.5%         25%         50%         75%
##     mu       80.05983 0.1042315    79.84607    79.98899    80.05924    80.13421
##     sigma    21.47386 0.7318103    20.02341    20.97457    21.46446    21.98952
##     lp__  -1548.43115 1.0657246 -1551.15451 -1548.88952 -1548.07700 -1547.69296
##          stats
## parameter       97.5%
##     mu       80.24876
##     sigma    22.87531
##     lp__  -1547.39427
##
## , , chains = chain:4
##
##          stats
## parameter        mean         sd        2.5%         25%         50%
##     mu       80.06190 0.09777108    79.86706    79.99749    80.06243
##     sigma    21.39178 0.72931090    20.07194    20.87985    21.38501
```

```
##      lp__  -1548.36056 0.95464328 -1551.09338 -1548.73999 -1548.08493
##           stats
## parameter         75%       97.5%
##      mu       80.12230    80.26466
##      sigma    21.86829    22.89559
##      lp__  -1547.64010 -1547.38979
```

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

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")
```



4 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 = "/Users/siyiwei/Desktop/applied-stats-2021/code/models/kids3.stan",
            data = data,
            iter = 1000)
```

4.1 Question 3

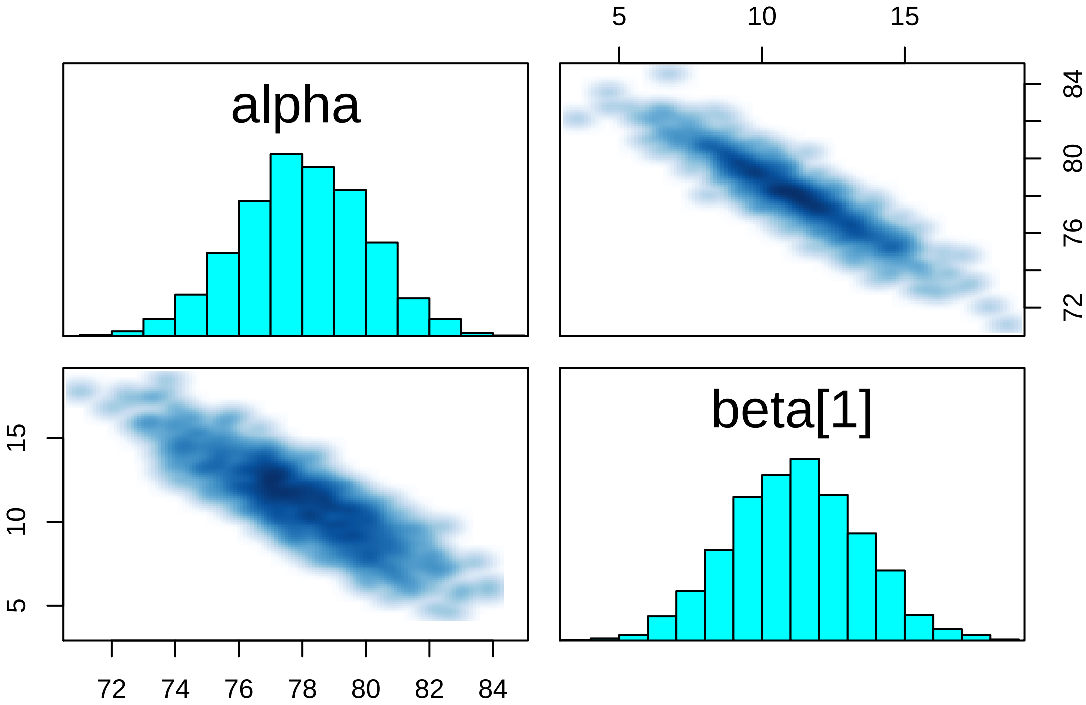
a) Confirm that the estimates of the intercept and slope are comparable to results from `lm()`

Answer: From the estimation of Stan model we could conclude alpha to be 78.02 and the estimation of beta to be 11.18. From the linear model we could conclude a similar result such that intercept to be 77.54 and slope to be 11.77.

b) Do a `pairs` plot to investigate the joint sample distributions of the slope and intercept. Comment briefly on what you see. Is this potentially a problem?

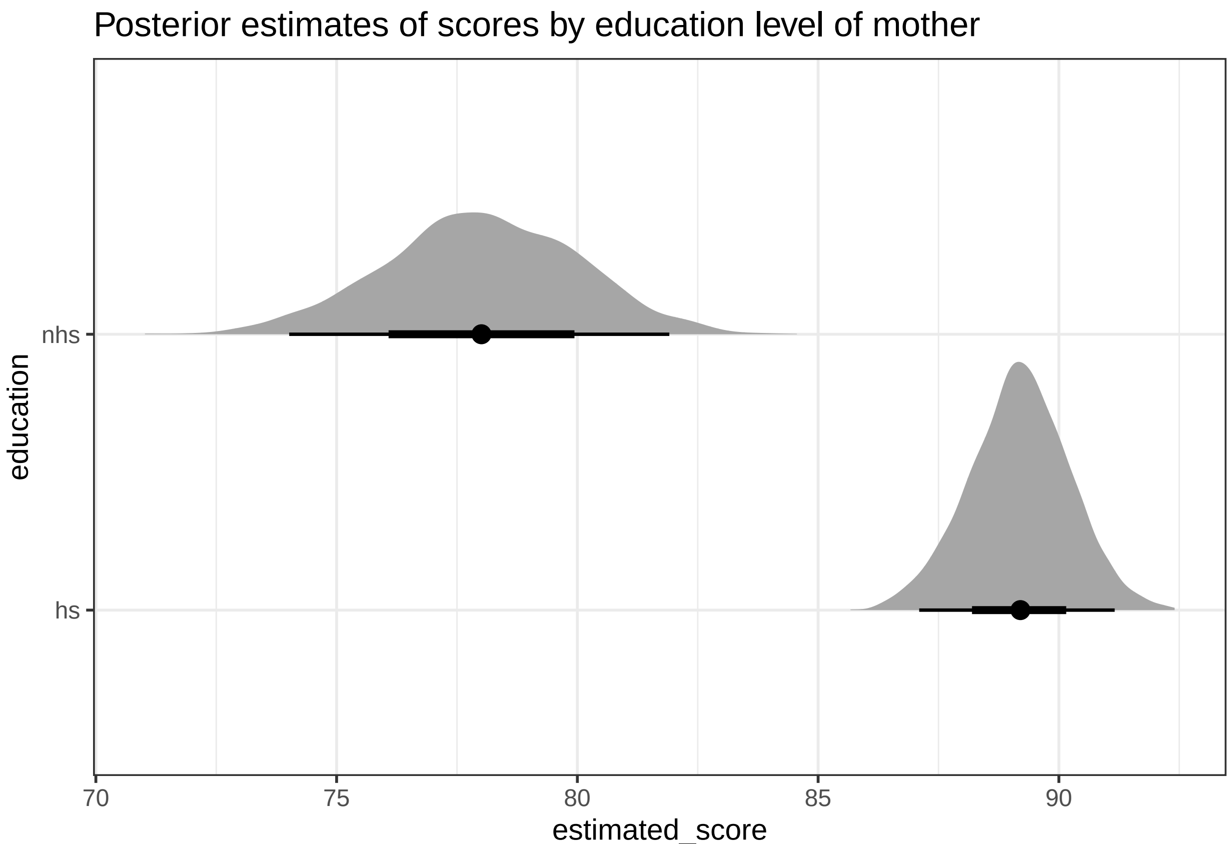
Answer: They have a very strong linear correlation. Which could potentially reveal collinearity between alpha and beta. It is not good for our estimation for sure since they are not randomly distributed.

```
fit2

lm_model <- lm(kid_score ~ mom_hs, data = kidiq)
print(lm_model$coefficients)

pars = c("alpha", "beta[1]")
pairs(fit2, pars = pars)
```



4.2 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



4.3 Question 4

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

Answer:

- After the center of the covariates. From Stan we could get the alpha to be 82.30, the mom_hs to be 5.72 and the mom iq to be 0.56
- From the coefficient. We could interpret as each unit of mum's IQ increase. It could improve the kid_score by 0.56 unit.

```
X <- as.matrix(cbind(kidiq$mom_hs, kidiq$mom_iq - mean(kidiq$mom_iq)), ncol = 2)
K <- 2

data <- list(y = y, N = length(y),
             X =X, K = K)
fit2 <- stan(file = "/Users/siyiwei/Desktop/applied-stats-2021/code/models/kids3.stan",
            data = data,
            iter = 1000)
```

4.4 Question 5

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

Answer:

- For the linear model. We could conclude the same result. The alpha is 86.79, the coefficient for mon_hs is 5.95 and for mom_iq is 0.563.

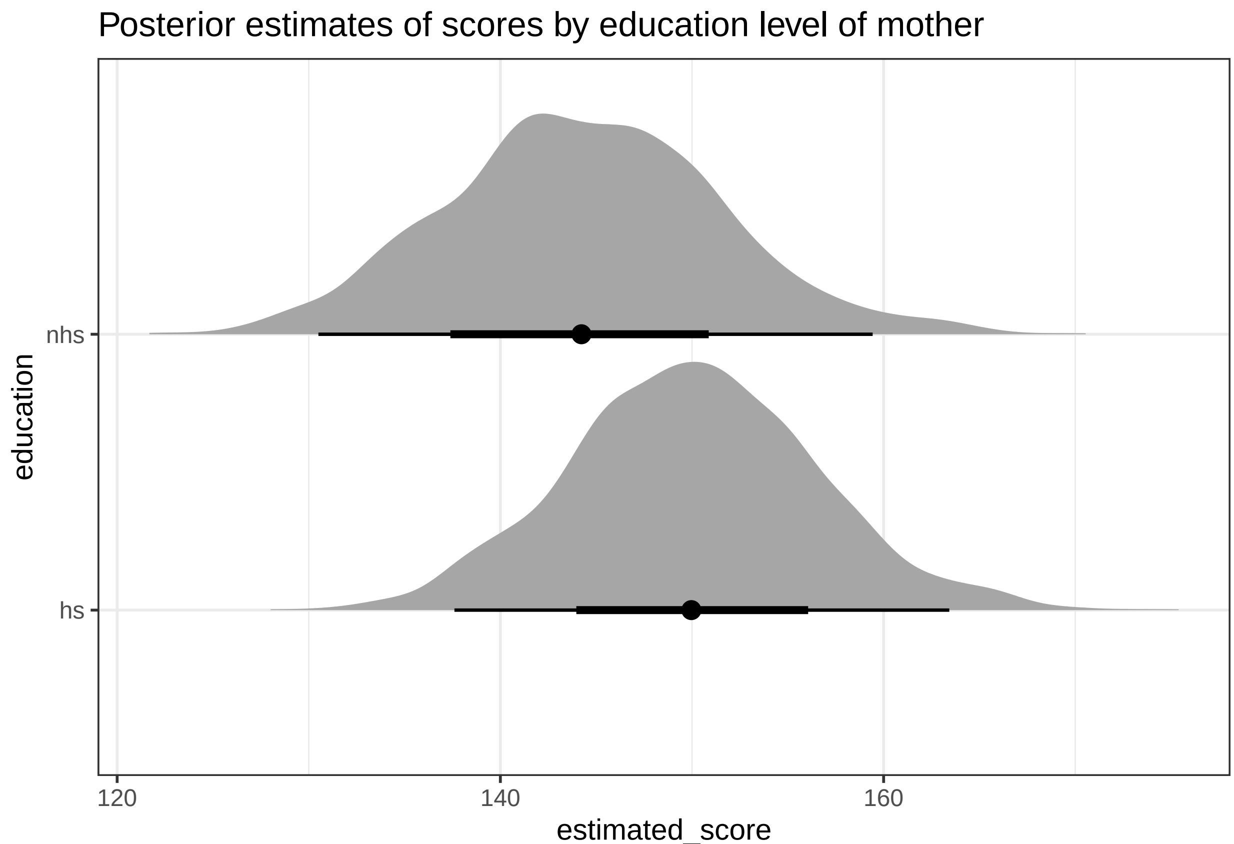
```
##
## Call:
## lm(formula = kid_score ~ mom_hs + mom_iq, data = kidiq2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.873 -12.663   2.404  11.356  49.545
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 86.79724    0.87054  99.705  < 2e-16 ***
## 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
```

4.5 Question 6

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

Answer: The plot is shown below.

```
library(purrr)
fit2 %>%
  spread_draws(alpha, beta[1], beta[2], sigma) %>%
     mutate(nhs = alpha + as.numeric(map(beta,2))*110,
          hs = alpha + as.numeric(map(beta,1)) + as.numeric(map(beta,2))*110) %>%
  pivot_longer(nhs:hs, names_to = "education", values_to = "estimated_score") %>%
  ggplot(aes(y = education, x = estimated_score)) +
  stat_halfeye() +
  theme_bw() +
  ggtitle("Posterior estimates of scores by education level of mother")
```



Posterior estimates of scores by education level of mother
nhs
hs
education
120
140
160
estimated_score