

Lab 6

```
library(tidyverse)
library(here)
# for bayes stuff
library(rstan)
library(bayesplot)
library(loo)
library(tidybayes)
```

```
ds <- read_rds(here("data","births_2017_sample.RDS"))
head(ds)
```

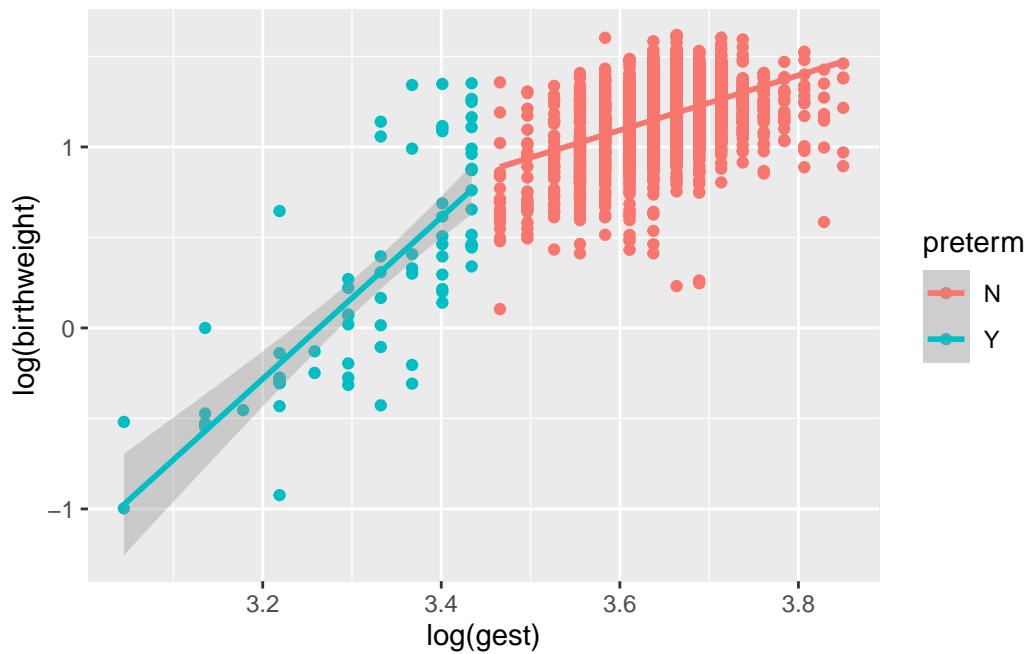
```
# A tibble: 6 x 8
  mager mracehisp meduc   bmi sex   combgest   dbwt ilive
  <dbl>      <dbl> <dbl> <dbl> <chr>    <dbl> <dbl> <chr>
1    16         2    2  23    M        39  3.18 Y
2    25         7    2 43.6 M        40  4.14 Y
3    27         2    3 19.5 F        41  3.18 Y
4    26         1    3 21.5 F        36  3.40 Y
5    28         7    2 40.6 F        34  2.71 Y
6    31         7    3 29.3 M        35  3.52 Y
```

```
ds <- ds %>%
  rename(birthweight = dbwt, gest = combgest) %>%
  mutate(preterm = ifelse(gest<32, "Y", "N")) %>%
  filter(ilive=="Y",gest< 99, birthweight<9.999)
```

Question 1

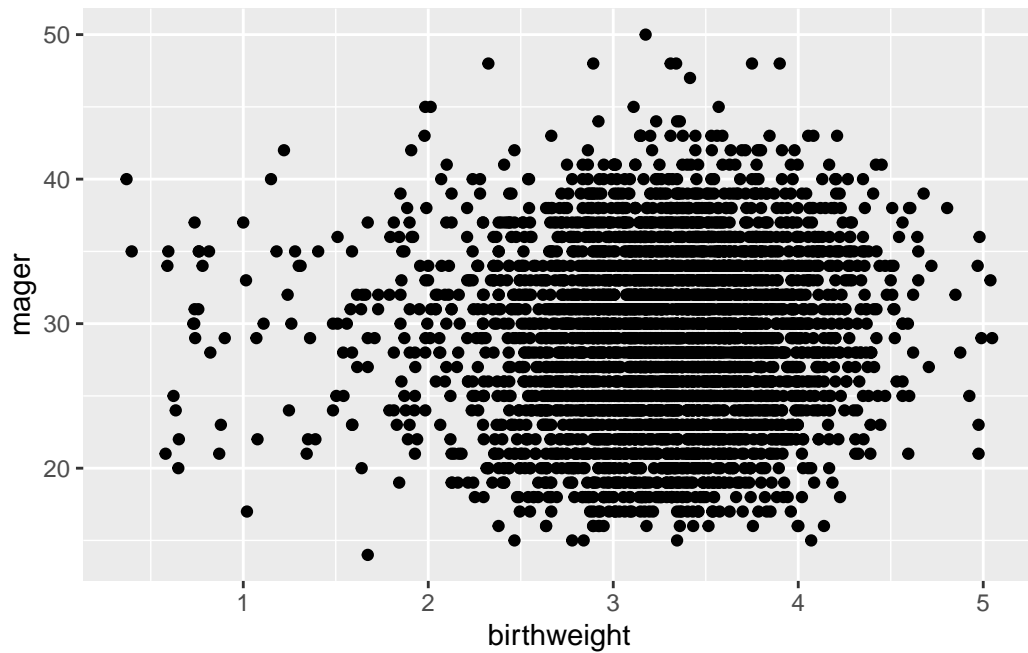
```
ggplot(ds,aes(x=log(gest),y=log(birthweight),color = preterm))+geom_point()+geom_smooth(me
```

``geom_smooth()`` using formula = `'y ~ x'`



The first interest point is that as gestational age increases, the infant's weight increases, and in the preterm period, the baby's body weight increases more rapidly.

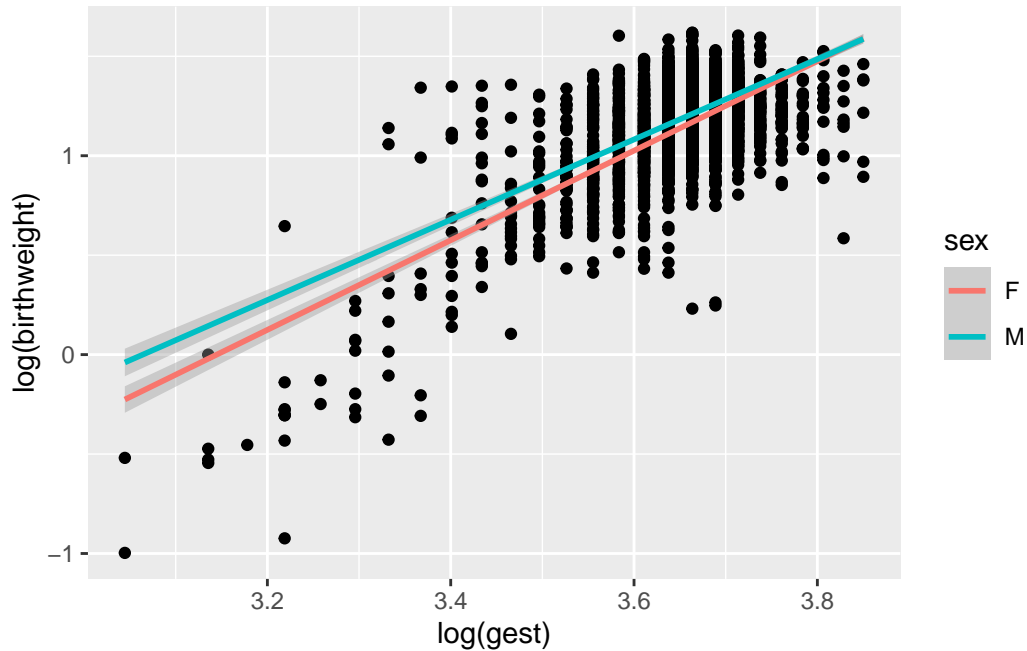
```
ggplot(ds,aes(x=birthweight,y=mager))+geom_point()
```



The second interest point is that there does not seem to be a strong relationship between the baby's weight and the mother's age.

```
ggplot(ds,aes(x=log(gest),y=log(birthweight)))+geom_point()+geom_smooth(method=lm,aes(color=
```

```
`geom_smooth()` using formula = 'y ~ x'
```



The third interest point is that as the gestational age increases, the birth weight increase for both baby boys and baby girls, and increase rate is slightly higher in baby girls than baby boys.

Question 2

```
set.seed(100)
nsims <- 1000
sigma <- rnorm(nsims, 0, 1)
beta1 <- rnorm(nsims, 0, 1)
beta2 <- rnorm(nsims, 0, 1)

dsims <- tibble(log_gest_c = (log(ds$gest)-mean(log(ds$gest)))/sd(log(ds$gest)))

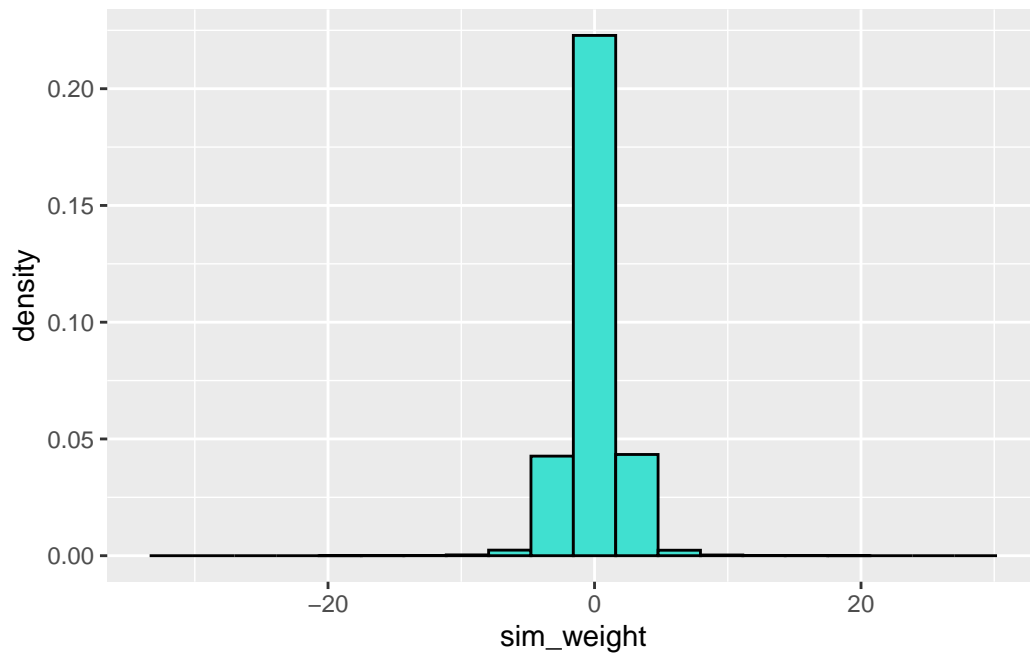
for(i in 1:nsims){
  this_mu <- beta1[i] + beta2[i]*dsims$log_gest_c
  dsims[paste0(i)] <- this_mu + rnorm(nrow(dsims), 0, sigma[i])
}

dsl <- dsims %>%
  pivot_longer(`1`:`1000`, names_to = "sim", values_to = "sim_weight")
```

```

dsl %>%
  ggplot(aes(sim_weight)) + geom_histogram(aes(y = ..density..), bins = 20, fill = "turquoise")

```

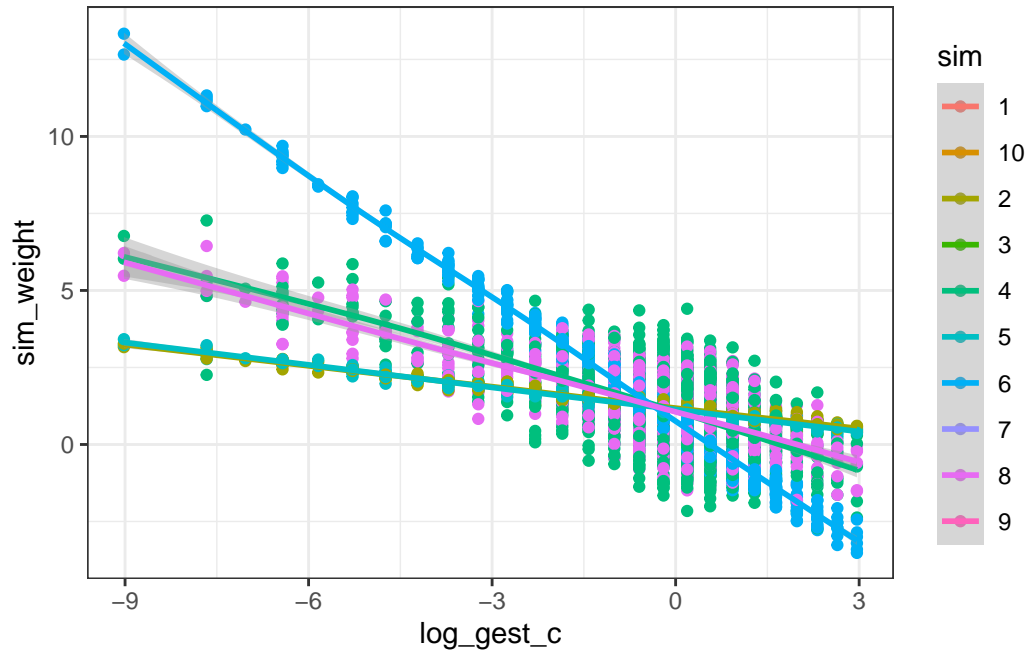


```

dsl10 <- dsims %>%
  pivot_longer(`1`:`10`, names_to = "sim", values_to = "sim_weight")
ggplot(dsl10, aes(x=log_gest_c, y=sim_weight, color=sim)) + geom_point() + geom_smooth() + theme_bw()

```

`geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'



Run the model

Now we're going to run Model 1 in Stan. The stan code is in the `code/models` folder.

First, get our data into right form for input into stan.

```
ds$log_weight <- log(ds$birthweight)
ds$log_gest_c <- (log(ds$gest) - mean(log(ds$gest)))/sd(log(ds$gest))

# put into a list
stan_data <- list(N = nrow(ds),
                  log_weight = ds$log_weight,
                  log_gest = ds$log_gest_c)
```

Now fit the model

```
mod1 <- stan(data = stan_data,
             file = here("code/models/simple_weight.stan"),
             iter = 500,
             seed = 243)
```

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).

Chain 1:

Chain 1: Gradient evaluation took 0.000334 seconds

Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 3.34 seconds.

Chain 1: Adjust your expectations accordingly!

Chain 1:

Chain 1:

Chain 1: Iteration: 1 / 500 [0%] (Warmup)

Chain 1: Iteration: 50 / 500 [10%] (Warmup)

Chain 1: Iteration: 100 / 500 [20%] (Warmup)

Chain 1: Iteration: 150 / 500 [30%] (Warmup)

Chain 1: Iteration: 200 / 500 [40%] (Warmup)

Chain 1: Iteration: 250 / 500 [50%] (Warmup)

Chain 1: Iteration: 251 / 500 [50%] (Sampling)

Chain 1: Iteration: 300 / 500 [60%] (Sampling)

Chain 1: Iteration: 350 / 500 [70%] (Sampling)

Chain 1: Iteration: 400 / 500 [80%] (Sampling)

Chain 1: Iteration: 450 / 500 [90%] (Sampling)

Chain 1: Iteration: 500 / 500 [100%] (Sampling)

Chain 1:

Chain 1: Elapsed Time: 0.359 seconds (Warm-up)

Chain 1: 0.301 seconds (Sampling)

Chain 1: 0.66 seconds (Total)

Chain 1:

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).

Chain 2:

Chain 2: Gradient evaluation took 0.000117 seconds

Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 1.17 seconds.

Chain 2: Adjust your expectations accordingly!

Chain 2:

Chain 2:

Chain 2: Iteration: 1 / 500 [0%] (Warmup)

Chain 2: Iteration: 50 / 500 [10%] (Warmup)

Chain 2: Iteration: 100 / 500 [20%] (Warmup)

Chain 2: Iteration: 150 / 500 [30%] (Warmup)

Chain 2: Iteration: 200 / 500 [40%] (Warmup)

Chain 2: Iteration: 250 / 500 [50%] (Warmup)

Chain 2: Iteration: 251 / 500 [50%] (Sampling)

Chain 2: Iteration: 300 / 500 [60%] (Sampling)

Chain 2: Iteration: 350 / 500 [70%] (Sampling)

Chain 2: Iteration: 400 / 500 [80%] (Sampling)

Chain 2: Iteration: 450 / 500 [90%] (Sampling)

Chain 2: Iteration: 500 / 500 [100%] (Sampling)
Chain 2:
Chain 2: Elapsed Time: 0.354 seconds (Warm-up)
Chain 2: 0.332 seconds (Sampling)
Chain 2: 0.686 seconds (Total)
Chain 2:

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).

Chain 3:
Chain 3: Gradient evaluation took 0.000115 seconds
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 1.15 seconds.
Chain 3: Adjust your expectations accordingly!
Chain 3:
Chain 3:
Chain 3: Iteration: 1 / 500 [0%] (Warmup)
Chain 3: Iteration: 50 / 500 [10%] (Warmup)
Chain 3: Iteration: 100 / 500 [20%] (Warmup)
Chain 3: Iteration: 150 / 500 [30%] (Warmup)
Chain 3: Iteration: 200 / 500 [40%] (Warmup)
Chain 3: Iteration: 250 / 500 [50%] (Warmup)
Chain 3: Iteration: 251 / 500 [50%] (Sampling)
Chain 3: Iteration: 300 / 500 [60%] (Sampling)
Chain 3: Iteration: 350 / 500 [70%] (Sampling)
Chain 3: Iteration: 400 / 500 [80%] (Sampling)
Chain 3: Iteration: 450 / 500 [90%] (Sampling)
Chain 3: Iteration: 500 / 500 [100%] (Sampling)
Chain 3:
Chain 3: Elapsed Time: 0.341 seconds (Warm-up)
Chain 3: 0.292 seconds (Sampling)
Chain 3: 0.633 seconds (Total)
Chain 3:

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).

Chain 4:
Chain 4: Gradient evaluation took 0.000126 seconds
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 1.26 seconds.
Chain 4: Adjust your expectations accordingly!
Chain 4:
Chain 4:
Chain 4: Iteration: 1 / 500 [0%] (Warmup)
Chain 4: Iteration: 50 / 500 [10%] (Warmup)
Chain 4: Iteration: 100 / 500 [20%] (Warmup)
Chain 4: Iteration: 150 / 500 [30%] (Warmup)


```
Chain 4: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 4: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 4: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 4: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 4: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 4: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 4: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 4: Iteration: 500 / 500 [100%] (Sampling)
Chain 4:
Chain 4: Elapsed Time: 0.352 seconds (Warm-up)
Chain 4:                0.292 seconds (Sampling)
Chain 4:                0.644 seconds (Total)
Chain 4:
```

Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be unreliable. Running the chains for more iterations may help. See <https://mc-stan.org/misc/warnings.html#bulk-ess>

Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quantiles may be unreliable. Running the chains for more iterations may help. See <https://mc-stan.org/misc/warnings.html#tail-ess>

```
summary(mod1)$summary[c("beta[1]", "beta[2]", "sigma"),]
```

	mean	se_mean	sd	2.5%	25%	50%
beta[1]	1.1624783	8.160385e-05	0.002856578	1.1570200	1.1604786	1.1625011
beta[2]	0.1437529	8.295075e-05	0.002912236	0.1381284	0.1416970	0.1436747
sigma	0.1690330	1.113724e-04	0.001902828	0.1652694	0.1677842	0.1690763

	75%	97.5%	n_eff	Rhat
beta[1]	1.1644669	1.1681028	1225.3801	0.9978044
beta[2]	0.1456716	0.1495180	1232.5721	0.9998714
sigma	0.1702528	0.1727953	291.9066	1.0146111

Question 3

```
est_37 <- (log(37)-mean(log(ds$gest)))/sd(log(ds$gest))
est_beta1 <- 1.1624783
est_beta2 <- 0.1437529
exp(est_beta1+est_beta2*est_37)
```

[1] 2.935874

Question 4

```
stan_data_mod2 <- list(N = nrow(ds),  
  log_weight = ds$log_weight,  
  log_gest = ds$log_gest_c,  
  preterm = ifelse(ds$preterm=="Y",1,0))  
  
mod2 <- stan(data = stan_data_mod2,  
  file = "Lab6mod2.stan",  
  iter = 500,  
  seed = 243)
```

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).

Chain 1:

Chain 1: Gradient evaluation took 0.000759 seconds

Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 7.59 seconds.

Chain 1: Adjust your expectations accordingly!

Chain 1:

Chain 1:

Chain 1: Iteration: 1 / 500 [0%] (Warmup)

Chain 1: Iteration: 50 / 500 [10%] (Warmup)

Chain 1: Iteration: 100 / 500 [20%] (Warmup)

Chain 1: Iteration: 150 / 500 [30%] (Warmup)

Chain 1: Iteration: 200 / 500 [40%] (Warmup)

Chain 1: Iteration: 250 / 500 [50%] (Warmup)

Chain 1: Iteration: 251 / 500 [50%] (Sampling)

Chain 1: Iteration: 300 / 500 [60%] (Sampling)

Chain 1: Iteration: 350 / 500 [70%] (Sampling)

Chain 1: Iteration: 400 / 500 [80%] (Sampling)

Chain 1: Iteration: 450 / 500 [90%] (Sampling)

Chain 1: Iteration: 500 / 500 [100%] (Sampling)

Chain 1:

Chain 1: Elapsed Time: 1.368 seconds (Warm-up)

Chain 1: 1.135 seconds (Sampling)

Chain 1: 2.503 seconds (Total)

Chain 1:

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).

```

Chain 2:
Chain 2: Gradient evaluation took 0.000287 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 2.87 seconds.
Chain 2: Adjust your expectations accordingly!
Chain 2:
Chain 2:
Chain 2: Iteration:   1 / 500 [  0%] (Warmup)
Chain 2: Iteration:  50 / 500 [ 10%] (Warmup)
Chain 2: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 2: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 2: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 2: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 2: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 2: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 2: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 2: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 2: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 2: Iteration: 500 / 500 [100%] (Sampling)
Chain 2:
Chain 2: Elapsed Time: 1.318 seconds (Warm-up)
Chain 2:                  1.255 seconds (Sampling)
Chain 2:                  2.573 seconds (Total)
Chain 2:

```

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).

```

Chain 3:
Chain 3: Gradient evaluation took 0.000317 seconds
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 3.17 seconds.
Chain 3: Adjust your expectations accordingly!
Chain 3:
Chain 3:
Chain 3: Iteration:   1 / 500 [  0%] (Warmup)
Chain 3: Iteration:  50 / 500 [ 10%] (Warmup)
Chain 3: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 3: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 3: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 3: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 3: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 3: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 3: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 3: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 3: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 3: Iteration: 500 / 500 [100%] (Sampling)

```

```

Chain 3:
Chain 3: Elapsed Time: 1.383 seconds (Warm-up)
Chain 3:           1.142 seconds (Sampling)
Chain 3:           2.525 seconds (Total)
Chain 3:

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
Chain 4:
Chain 4: Gradient evaluation took 0.000327 seconds
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 3.27 seconds.
Chain 4: Adjust your expectations accordingly!
Chain 4:
Chain 4:
Chain 4: Iteration:   1 / 500 [  0%] (Warmup)
Chain 4: Iteration:  50 / 500 [ 10%] (Warmup)
Chain 4: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 4: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 4: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 4: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 4: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 4: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 4: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 4: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 4: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 4: Iteration: 500 / 500 [100%] (Sampling)
Chain 4:
Chain 4: Elapsed Time: 1.403 seconds (Warm-up)
Chain 4:           1.27 seconds (Sampling)
Chain 4:           2.673 seconds (Total)
Chain 4:

```

```
summary(mod2)$summary[c("beta[1]", "beta[2]", "beta[3]", "beta[4]", "sigma"),]
```

	mean	se_mean	sd	2.5%	25%	50%
beta[1]	1.1696329	8.021297e-05	0.002705139	1.16410383	1.16791913	1.1695478
beta[2]	0.1018545	1.111662e-04	0.003424916	0.09508969	0.09961365	0.1019319
beta[3]	0.5620695	3.406560e-03	0.062560942	0.43112646	0.52265217	0.5614275
beta[4]	0.1982641	6.964438e-04	0.012807594	0.17144797	0.18979854	0.1986269
sigma	0.1611971	8.785429e-05	0.001825790	0.15774991	0.15994557	0.1611909
	75%	97.5%	n_eff	Rhat		
beta[1]	1.1714725	1.1748162	1137.3388	1.000638		

```

beta[2] 0.1040358 0.1087724 949.1923 1.002232
beta[3] 0.6039584 0.6839901 337.2675 1.015352
beta[4] 0.2062635 0.2232005 338.1917 1.013325
sigma   0.1623667 0.1649513 431.8927 1.004553

```

Question 5

```

load(here("output", "mod2.Rda"))
summary(mod2)$summary[c(paste0("beta[", 1:4, "]"), "sigma"),]

```

	mean	se_mean	sd	2.5%	25%	50%
beta[1]	1.1697241	1.385590e-04	0.002742186	1.16453578	1.16767109	1.1699278
beta[2]	0.5563133	5.835253e-03	0.058054991	0.43745504	0.51708255	0.5561553
beta[3]	0.1020960	1.481816e-04	0.003669476	0.09459462	0.09997153	0.1020339
beta[4]	0.1967671	1.129799e-03	0.012458398	0.17164533	0.18817091	0.1974114
sigma	0.1610727	9.950037e-05	0.001782004	0.15784213	0.15978020	0.1610734

	75%	97.5%	n_eff	Rhat
beta[1]	1.1716235	1.1750167	391.67359	1.0115970
beta[2]	0.5990427	0.6554967	98.98279	1.0088166
beta[3]	0.1044230	0.1093843	613.22428	0.9978156
beta[4]	0.2064079	0.2182454	121.59685	1.0056875
sigma	0.1623019	0.1646189	320.75100	1.0104805

Based on the summary, the coefficients are similar after flipping beta[2] and beta[3].

Question 6

```

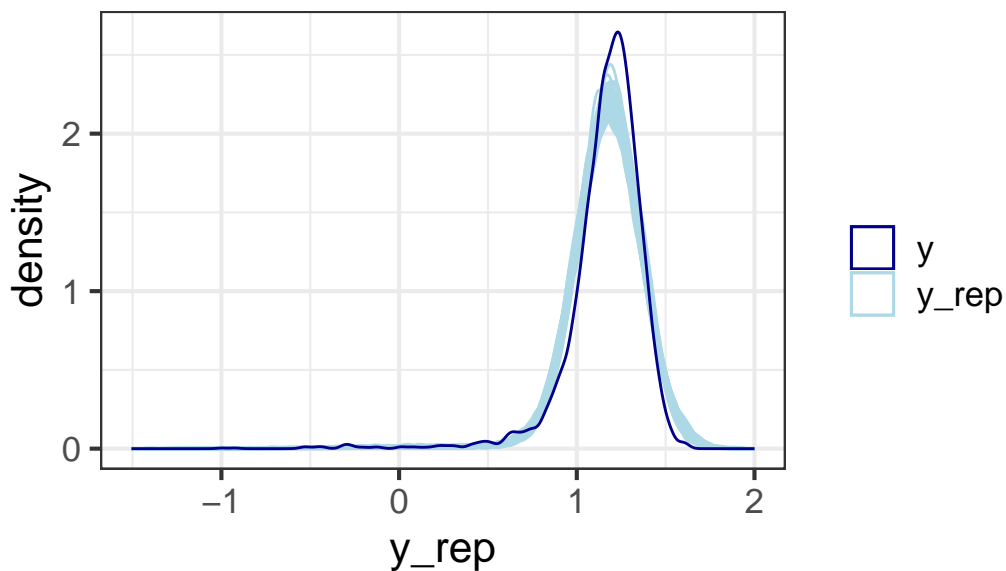
N = nrow(ds)
yrep2 <- extract(mod2)[['log_weight_rep']]
samp2 <- sample(nrow(yrep2),100)
# first, get into a tibble
rownames(yrep2) <- 1:nrow(yrep2)
dr <- as_tibble(t(yrep2))
dr <- dr %>% bind_cols(i = 1:N, log_weight_obs = log(ds$birthweight))

# turn into long format; easier to plot
dr <- dr %>%
  pivot_longer(-(i:log_weight_obs), names_to = "sim", values_to = "y_rep")

```

```
# filter to just include 100 draws and plot!
dr %>%
  filter(sim %in% samp2) %>%
  ggplot(aes(y_rep, group = sim)) +
  geom_density(alpha = 0.2, aes(color = "y_rep")) +
  geom_density(data = ds %>% mutate(sim = 1),
               aes(x = log(birthweight), col = "y")) +
  scale_color_manual(name = "",
                    values = c("y" = "darkblue",
                              "y_rep" = "lightblue")) +
  ggtitle("Distribution of observed and replicated birthweights") +
  theme_bw(base_size = 16)
```

Distribution of observed and replicated birt



Question 7

```
yrep1 <- extract(mod1)[["log_weight_rep"]]

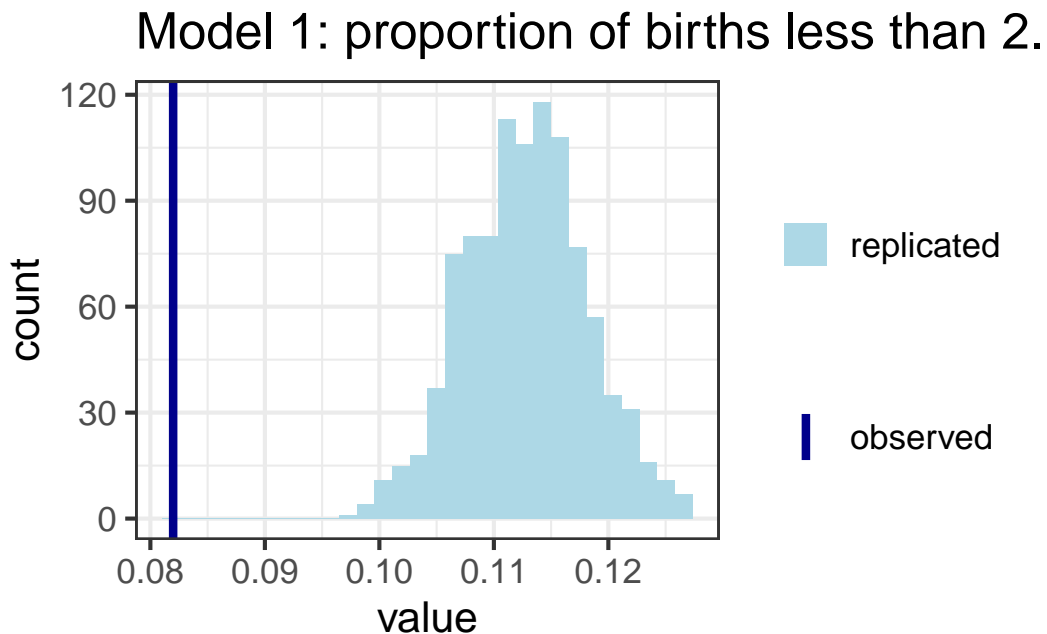
y <- ds$log_weight
t_y <- mean(y<=log(2.5))
t_y_rep <- sapply(1:nrow(yrep1), function(i) mean(yrep1[i,]<=log(2.5)))
```

```
t_y_rep_2 <- sapply(1:nrow(yrep2), function(i) mean(yrep2[i,]<=log(2.5)))
```

```
ggplot(data = as_tibble(t_y_rep), aes(value)) +
  geom_histogram(aes(fill = "replicated")) +
  geom_vline(aes(xintercept = t_y, color = "observed"), lwd = 1.5) +
  ggtitle("Model 1: proportion of births less than 2.5kg") +
  theme_bw(base_size = 16) +
  scale_color_manual(name = "",
                    values = c("observed" = "darkblue"))+
  scale_fill_manual(name = "",
                  values = c("replicated" = "lightblue"))
```

Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.

`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.



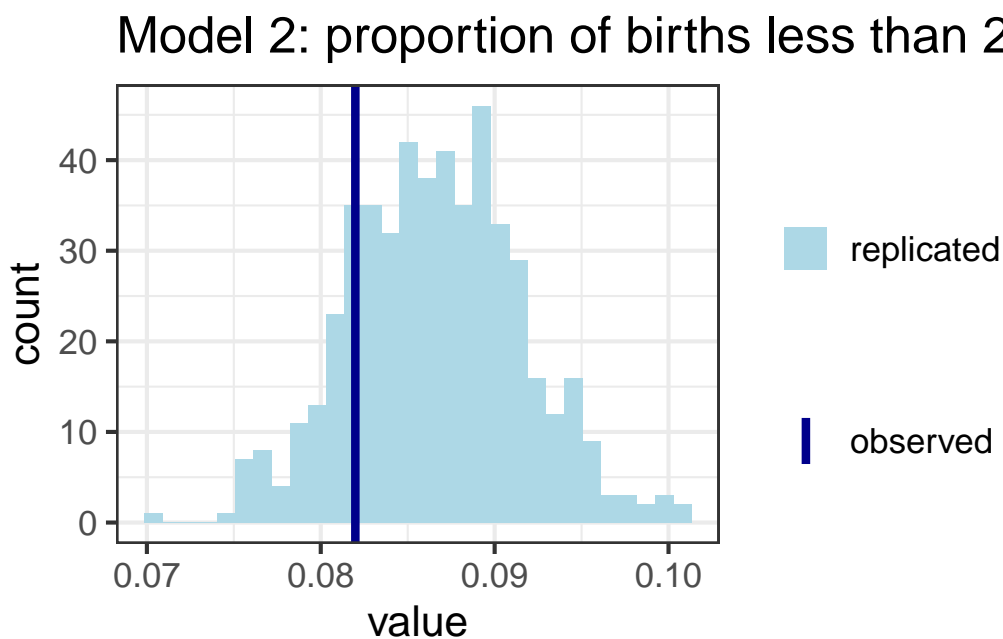
```
ggplot(data = as_tibble(t_y_rep_2), aes(value)) +
  geom_histogram(aes(fill = "replicated")) +
```

```

    geom_vline(aes(xintercept = t_y, color = "observed"), lwd = 1.5) +
    ggtitle("Model 2: proportion of births less than 2.5kg") +
    theme_bw(base_size = 16) +
    scale_color_manual(name = "",
                      values = c("observed" = "darkblue"))+
    scale_fill_manual(name = "",
                     values = c("replicated" = "lightblue"))

```

`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.



Question 8

Based on the EDA, add sex as an additional covariates to the linear regression model. Then, the model become:

$$\log(y_i) \sim N(\beta_1 + \beta_2 \log(x_i) + \beta_3 s_i, \sigma^2)$$

where

- y_i is weight in kg
- x_i is gestational age in weeks, CENTERED AND STANDARDIZED
- s_i is sex (0 for female and 1 for male)


```

stan_data_mod3 <- list(N = nrow(ds),
                      log_weight = ds$log_weight,
                      log_gest = ds$log_gest_c,
                      sex = ifelse(ds$sex=="M",1,0))
mod3 <- stan(data = stan_data_mod3,
             file = "Lab6mod3.stan",
             iter = 500,
             seed = 243)

```

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).

Chain 1:

Chain 1: Gradient evaluation took 0.000405 seconds

Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 4.05 seconds.

Chain 1: Adjust your expectations accordingly!

Chain 1:

Chain 1:

Chain 1: Iteration: 1 / 500 [0%] (Warmup)

Chain 1: Iteration: 50 / 500 [10%] (Warmup)

Chain 1: Iteration: 100 / 500 [20%] (Warmup)

Chain 1: Iteration: 150 / 500 [30%] (Warmup)

Chain 1: Iteration: 200 / 500 [40%] (Warmup)

Chain 1: Iteration: 250 / 500 [50%] (Warmup)

Chain 1: Iteration: 251 / 500 [50%] (Sampling)

Chain 1: Iteration: 300 / 500 [60%] (Sampling)

Chain 1: Iteration: 350 / 500 [70%] (Sampling)

Chain 1: Iteration: 400 / 500 [80%] (Sampling)

Chain 1: Iteration: 450 / 500 [90%] (Sampling)

Chain 1: Iteration: 500 / 500 [100%] (Sampling)

Chain 1:

Chain 1: Elapsed Time: 0.76 seconds (Warm-up)

Chain 1: 0.649 seconds (Sampling)

Chain 1: 1.409 seconds (Total)

Chain 1:

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 2).

Chain 2:

Chain 2: Gradient evaluation took 0.00026 seconds

Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 2.6 seconds.

Chain 2: Adjust your expectations accordingly!

Chain 2:

Chain 2:

```

Chain 2: Iteration: 1 / 500 [ 0%] (Warmup)
Chain 2: Iteration: 50 / 500 [ 10%] (Warmup)
Chain 2: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 2: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 2: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 2: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 2: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 2: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 2: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 2: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 2: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 2: Iteration: 500 / 500 [100%] (Sampling)
Chain 2:
Chain 2: Elapsed Time: 0.708 seconds (Warm-up)
Chain 2: 0.71 seconds (Sampling)
Chain 2: 1.418 seconds (Total)
Chain 2:

```

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 3).

```

Chain 3:
Chain 3: Gradient evaluation took 0.000256 seconds
Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 2.56 seconds.
Chain 3: Adjust your expectations accordingly!
Chain 3:
Chain 3:
Chain 3: Iteration: 1 / 500 [ 0%] (Warmup)
Chain 3: Iteration: 50 / 500 [ 10%] (Warmup)
Chain 3: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 3: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 3: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 3: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 3: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 3: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 3: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 3: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 3: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 3: Iteration: 500 / 500 [100%] (Sampling)
Chain 3:
Chain 3: Elapsed Time: 0.712 seconds (Warm-up)
Chain 3: 0.634 seconds (Sampling)
Chain 3: 1.346 seconds (Total)
Chain 3:

```

```

SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 4).
Chain 4:
Chain 4: Gradient evaluation took 0.000262 seconds
Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 2.62 seconds.
Chain 4: Adjust your expectations accordingly!
Chain 4:
Chain 4:
Chain 4: Iteration: 1 / 500 [ 0%] (Warmup)
Chain 4: Iteration: 50 / 500 [ 10%] (Warmup)
Chain 4: Iteration: 100 / 500 [ 20%] (Warmup)
Chain 4: Iteration: 150 / 500 [ 30%] (Warmup)
Chain 4: Iteration: 200 / 500 [ 40%] (Warmup)
Chain 4: Iteration: 250 / 500 [ 50%] (Warmup)
Chain 4: Iteration: 251 / 500 [ 50%] (Sampling)
Chain 4: Iteration: 300 / 500 [ 60%] (Sampling)
Chain 4: Iteration: 350 / 500 [ 70%] (Sampling)
Chain 4: Iteration: 400 / 500 [ 80%] (Sampling)
Chain 4: Iteration: 450 / 500 [ 90%] (Sampling)
Chain 4: Iteration: 500 / 500 [100%] (Sampling)
Chain 4:
Chain 4: Elapsed Time: 0.72 seconds (Warm-up)
Chain 4: 0.677 seconds (Sampling)
Chain 4: 1.397 seconds (Total)
Chain 4:

```

Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be unreliable. Running the chains for more iterations may help. See <https://mc-stan.org/misc/warnings.html#bulk-ess>

Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quantiles may be unreliable. Running the chains for more iterations may help. See <https://mc-stan.org/misc/warnings.html#tail-ess>

```
summary(mod3)$summary[c("beta[1]", "beta[2]", "beta[3]", "sigma"),]
```

	mean	se_mean	sd	2.5%	25%	50%
beta[1]	1.14028996	1.493403e-04	0.003754554	1.13285962	1.13779512	1.14018814
beta[2]	0.14414166	9.454262e-05	0.002773616	0.13860941	0.14230469	0.14404814
beta[3]	0.04436215	2.183887e-04	0.005392706	0.03392679	0.04069144	0.04438335
sigma	0.16739711	1.090949e-04	0.002013758	0.16359394	0.16609752	0.16731626
	75%	97.5%	n_eff	Rhat		

```

beta[1] 1.14287688 1.14764264 632.0665 0.9998937
beta[2] 0.14592509 0.15018203 860.6716 1.0007940
beta[3] 0.04802888 0.05480015 609.7522 1.0006674
sigma   0.16868116 0.17155964 340.7265 0.9974656

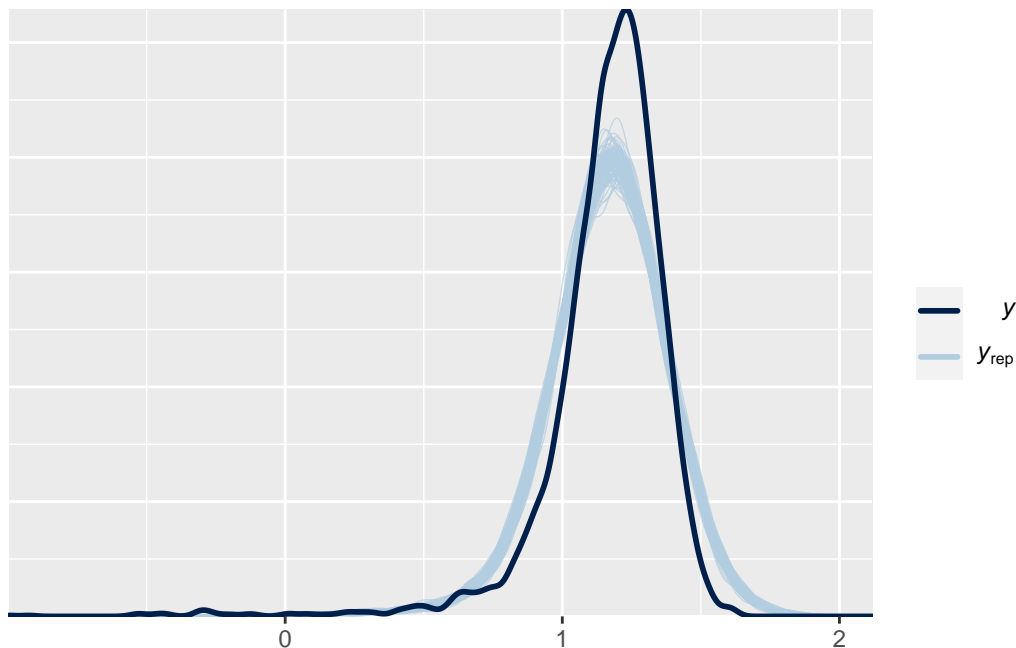
```

1. Comparing the 100 simulated datasets drawn from the posterior predictive distribution to the actual data.

```

yrep3 <- extract(mod3)[['log_weight_rep']]
samp3 <- sample(nrow(yrep3), 100)
ppc_dens_overlay(y, yrep3[samp3,])

```



2. Calculate the test statistic for the data, and the posterior predictive samples for both models, and plot the comparison.

```

t_y_rep_3 <- sapply(1:nrow(yrep3), function(i) mean(yrep3[i,]<=log(2.5)))
ggplot(data = as_tibble(t_y_rep_3), aes(value)) +
  geom_histogram(aes(fill = "replicated")) +
  geom_vline(aes(xintercept = t_y, color = "observed"), lwd = 1.5) +
  ggtitle("Model 3: proportion of births less than 2.5kg") +
  theme_bw(base_size = 16) +
  scale_color_manual(name = "",

```

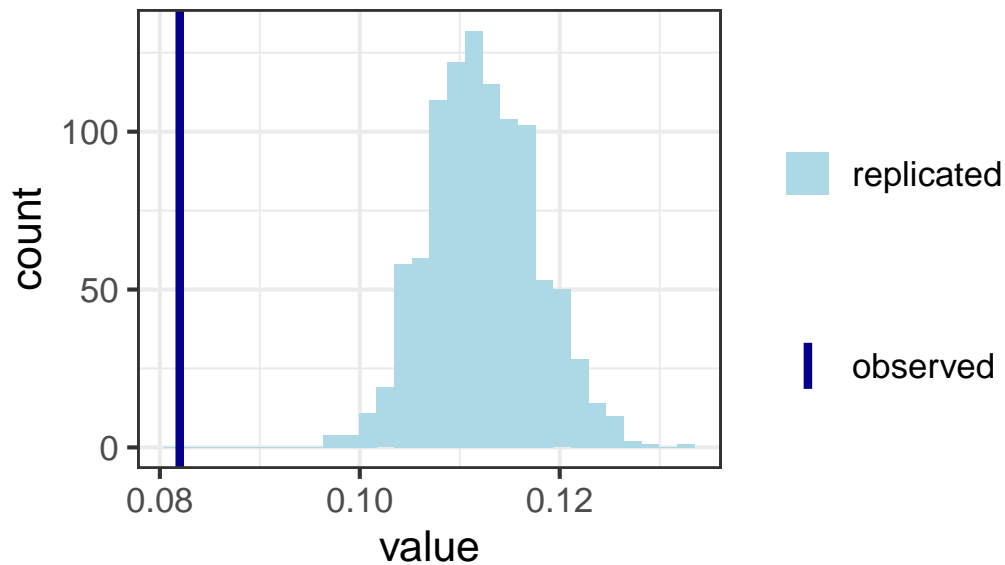
```

      values = c("observed" = "darkblue"))+
scale_fill_manual(name = "",
      values = c("replicated" = "lightblue"))

```

``stat_bin()`` using ``bins = 30``. Pick better value with ``binwidth``.

Model 3: proportion of births less than 2.



Thus, the result shows that comparing to model 3, model 2 is better.