Week 6: Visualizing the Bayesian Workflow

20/02/23

Introduction

This lab will be looking at trying to replicate some of the visualizations in the lecture notes, involving prior and posterior predictive checks, and LOO model comparisons.

The dataset is a 0.1% of all births in the US in 2017. I've pulled out a few different variables, but as in the lecture, we'll just focus on birth weight and gestational age.

The data

Read it in, along with all our packages.

```
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
```

Brief overview of variables:

- `mager` mum's age
- `mracehisp` mum's race/ethnicity see here for codes: https://data.nber.org/natality/2017/natl2017.pdf page 15
- `meduc` mum's education see here for codes: https://data.nber.org/natality/2017/natl2017.pdf page 16
- `bmi` mum's bmi
- `sex` baby's sex
- `combgest` gestational age in weeks
- `dbwt` birth weight in kg
- `ilive` alive at time of report y/n/ unsure

I'm going to rename some variables, remove any observations with missing gestational age or birth weight, restrict just to babies that were alive, and make a preterm variable.

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

Question 1

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

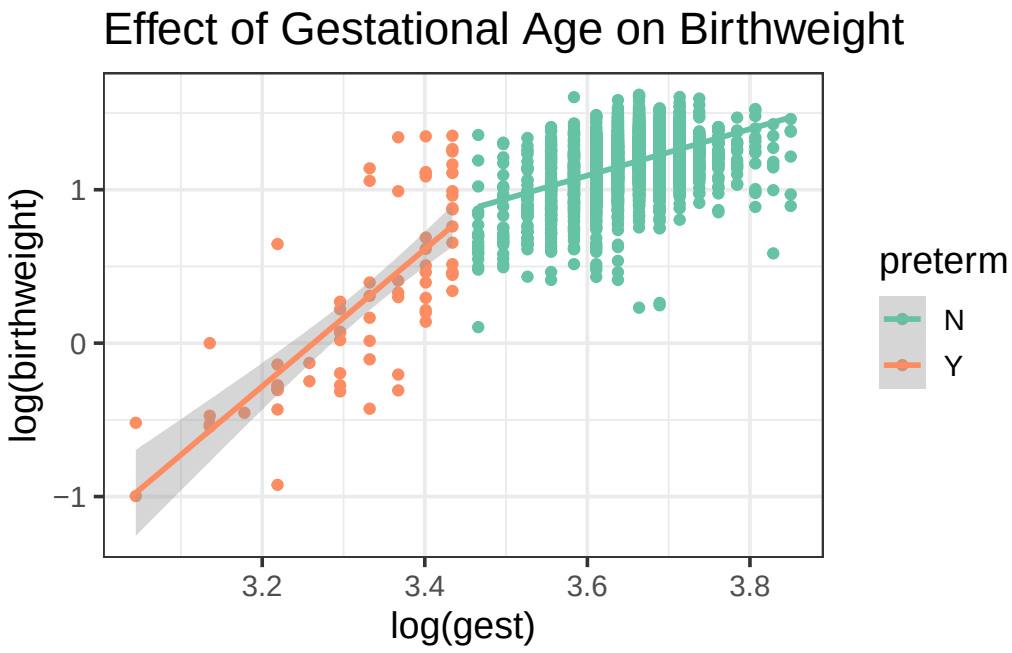
- Explain what your graph/ tables show
- Choose a graph type that's appropriate to the data type
- If you use `geom_smooth`, please also plot the underlying data

Feel free to replicate one of the scatter plots in the lectures as one of the interesting observations, as those form the basis of our models.

Let's first start by looking at how gestational age and birthweight are related. This is what we saw in class.

```
ds %>%
  ggplot(aes(log(gest), log(birthweight), color = preterm)) +
  geom_point() + geom_smooth(method = "lm") +
```

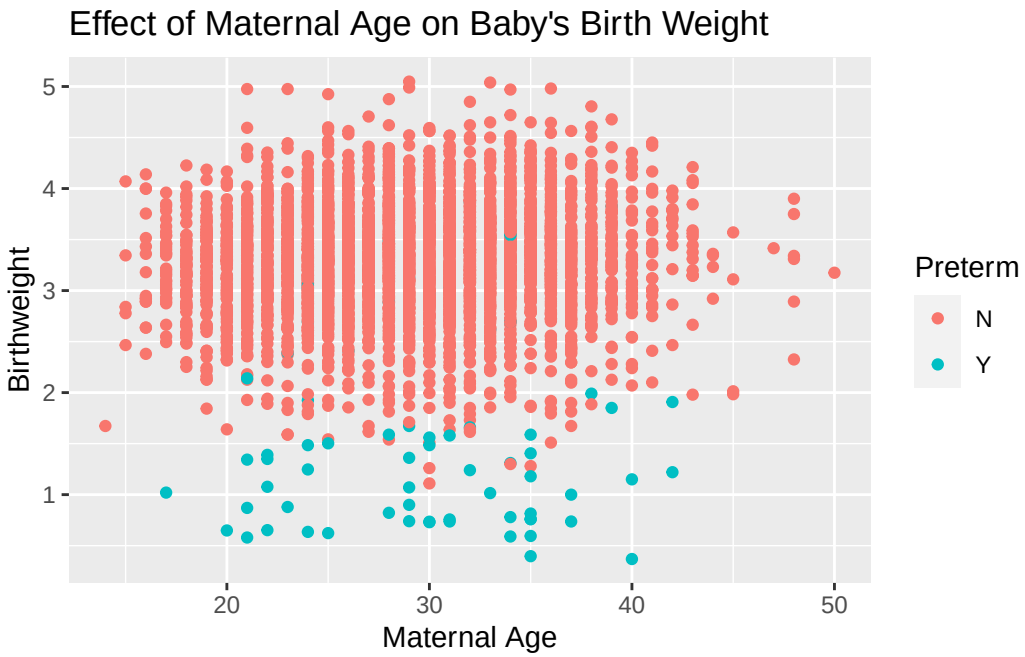
```
  scale_color_brewer(palette = "Set2") +
  theme_bw(base_size = 14) +
  ggtitle("Effect of Gestational Age on Birthweight")
```



We see that there are basiccally two regimes here, one for preterm babies and one for non-preterm babies. When preterm, there appears to be a linear relationship between the logarithm of gestation and the logarithm of birthweight. Thus, we will include the interaction between gestation and preterm status later.

It might also be good to analyze the relationship between maternal age and birthweight.

```
ggplot(ds, aes(x=mager, y=birthweight, color=preterm)) +
  geom_point() +
  labs(x="Maternal Age", y="Birthweight",
       title="Effect of Maternal Age on Baby's Birth Weight",
       color="Preterm")
```



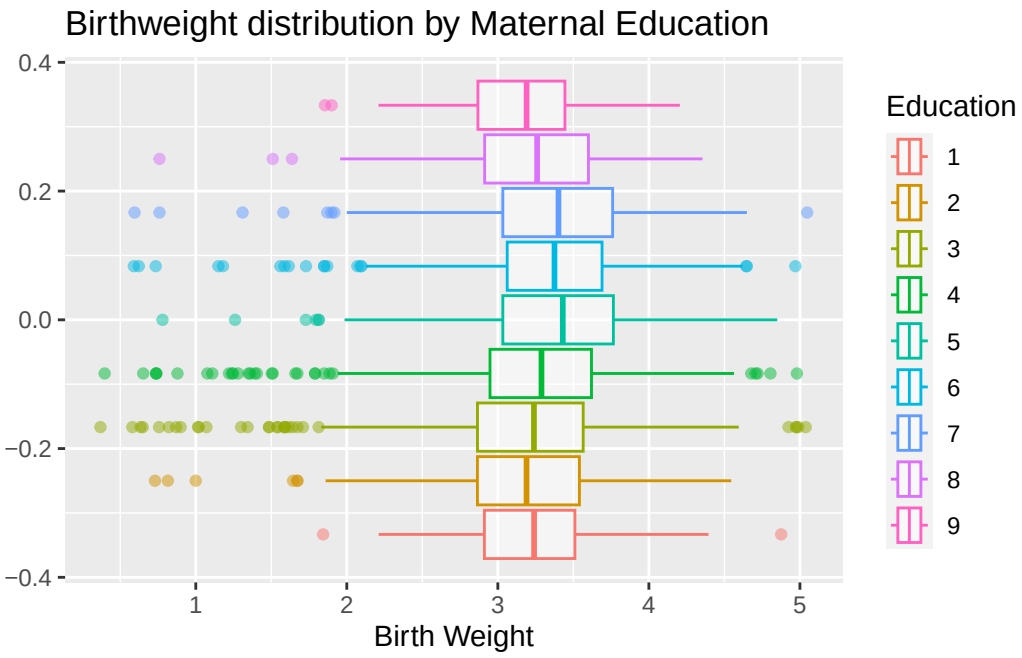
Clearly, preterm babies are typically lighter. But more interestingly, we have more data at middle ages. This makes sense as people less than 20 and over 40 are less likely to be new mothers. There does not seem to be any immediate relationship between age and birthweight, except for the fact that we have more data for certain age groups. It would be interesting to analyze this further, and I will do this in the last question.

We can also look for the relationship between birthweight and the `meduc` factor, the mother's education. This is categorical, so we can use a boxplot and color by group. We know that these can be grouped as:

1. 8th grade or less
2. 9th through 12th grade with no diploma
3. High school graduate or GED completed
4. Some college credit, but not a degree.
5. Associate degree (AA,AS)
6. Bachelor's degree (BA, AB, BS)
7. Master's degree (MA, MS, MEng, MEd, MSW, MBA)
8. Doctorate (PhD, EdD) or Professional Degree (MD, DDS, DVM, LLB, JD)
9. Unknown

```
ggplot(ds, aes(x=birthweight, color=as.factor(meduc))) +
  geom_boxplot(alpha = 0.5) +
  labs(x="Birth Weight",
```

```
        title="Birthweight distribution by Maternal Education",
        color="Education")
```



In our plot, the central tendency (median) does not change a lot between educational groups. The differences actually seem to occur at the extremes. People with 8th grade or less, or in the unknown group do not have many outliers. This is likely just because there are not many data points in general. There are lots of outliers for groups 4 and 5, but these possibly could also be the groups with the most data points. There might be an imbalance in data, so it is hard to make any definitive conclusions.

```
ds %>%
  group_by(meduc) %>%
  count()
```

```
# A tibble: 9 x 2
# Groups:   meduc [9]
  meduc     n
  <dbl> <int>
1     1   127
2     2   386
3     3   927
```

```
4      4   770
5      5   327
6      6   800
7      7   352
8      8   107
9      9    46
```

We can confirm that there are lots of discrepancies in the educational classes. Very few have professional degrees, and the majority have bachelors degrees or are high school/GED graduates. This imabalance could cause problems, and I will not include `meduc` as a covariate in my model in Q8.

The model

As in lecture, we will look at two candidate models

Model 1 has log birth weight as a function of log gestational age

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

Model 2 has an interaction term between gestation and prematurity

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

- y_i is weight in kg
- x_i is gestational age in weeks, CENTERED AND STANDARDIZED
- z_i is preterm (0 or 1, if gestational age is less than 32 weeks)

Prior predictive checks

Let's put some weakly informative priors on all parameters i.e. for the βs

$$\beta \sim N(0, 1)$$

and for σ

$$\sigma \sim N^+(0, 1)$$

where the plus means positive values only i.e. Half Normal.

Let's check to see what the resulting distribution of birth weights look like given Model 1 and the priors specified above, assuming we had no data on birth weight (but observations of gestational age).

Question 2

For Model 1, simulate values of βs and σ based on the priors above. Do 1000 simulations. Use these values to simulate (log) birth weights from the likelihood specified in Model 1, based on the set of observed gestational weights. **Remember the gestational weights should be centered and standardized**.

- Plot the resulting distribution of simulated (log) birth weights.
- Plot ten simulations of (log) birthweights against gestational age.

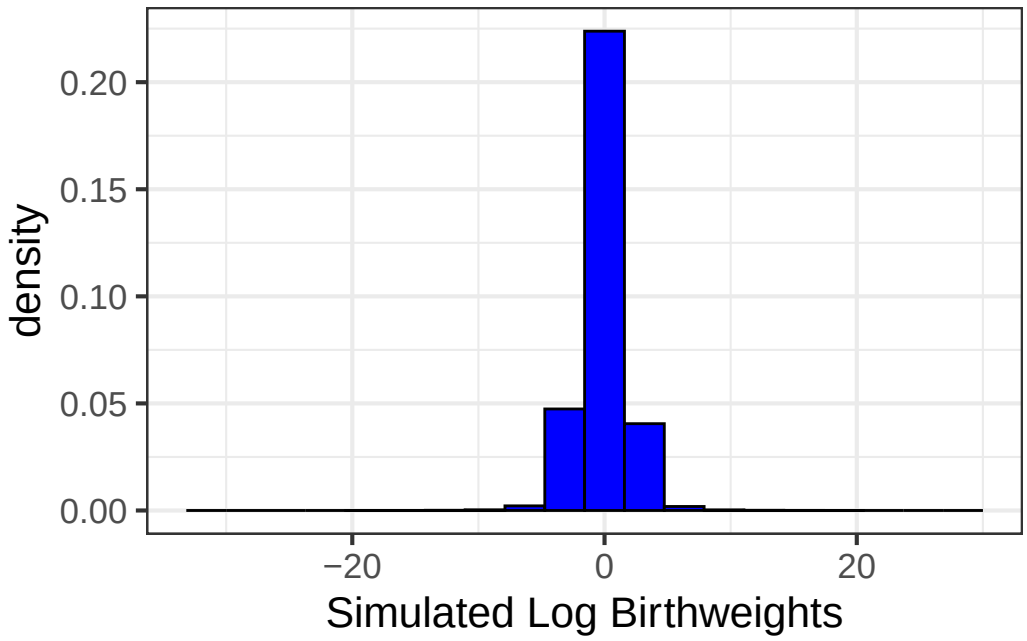
```
n_sims <- 1000
beta0 <- rnorm(n_sims, 0, 1)
beta1 <- rnorm(n_sims, 0, 1)
sigma <- abs(rnorm(n_sims, 0, 1))

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

for(i in 1:n_sims){
  this_mu <- beta0[i] + beta1[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 = "blue", color = "black") +
  xlab('Simulated Log Birthweights') +
  theme_bw(base_size = 16)
```

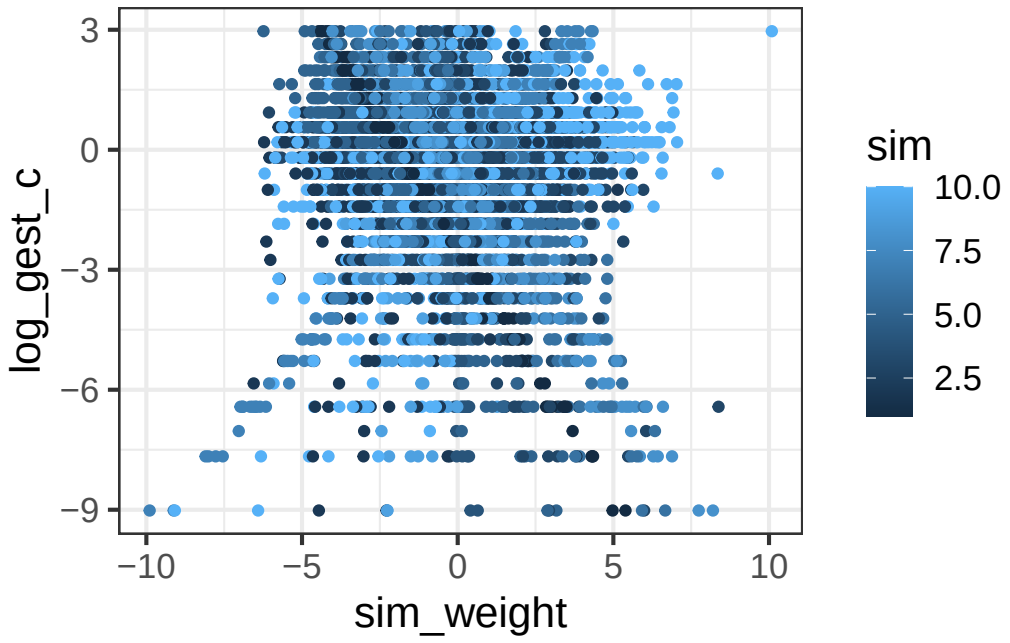


- Plot ten simulations of (log) birthweights against gestational age.

```
dsl <- dsims %>%
  pivot_longer(`1`:`10`, names_to = "sim", values_to = "sim_weight")

dsl$sim <- as.numeric(dsl$sim)

dsl %>%
  ggplot(aes(x=sim_weight, y=log_gest_c, color = sim)) + geom_point() +
  theme_bw(base_size = 16)
```



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)
```

```r
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
```

1 percent change in standardized version of our birth rate is eqivalent to a 14% change in the weight.

Question 3

Based on model 3, give an estimate of the expected birthweight of a baby who was born at a gestational age of 37 weeks. We must remember to exponentiate.

```r
beta1 <- summary(mod1)$summary[c("beta[1]"), "mean"]
beta2 <- summary(mod1)$summary[c("beta[2]"), "mean"]
lin_mode <- (log(37) - mean(log(ds$gest)))/sd(log(ds$gest))
exp(beta1 + beta2*(lin_mode))
```

```
[1] 2.935874
```

Therefore, we have an estimate for the expected birthweight of a baby born at gestational age of 37 weeks is around 2.935 kg.

Question 4

Write a stan model to run Model 2, and run it.

```r
ds$preterm_int <- ifelse(ds$preterm=='Y', 1, 0)

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

```
                  log_gest = ds$log_gest_c,
                  preterm = ds$preterm_int)
```

Now fit the model

```
mod2 <- stan(data = stan_data,
             file = here("code/models/simple_weight_preterm.stan"),
             iter = 500,
             seed = 243)
```

```
summary(mod2)$summary[c(paste0("beta[", 1:4, "]"), "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

For reference I have uploaded some model 2 results. Check your results are similar.

```
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
```

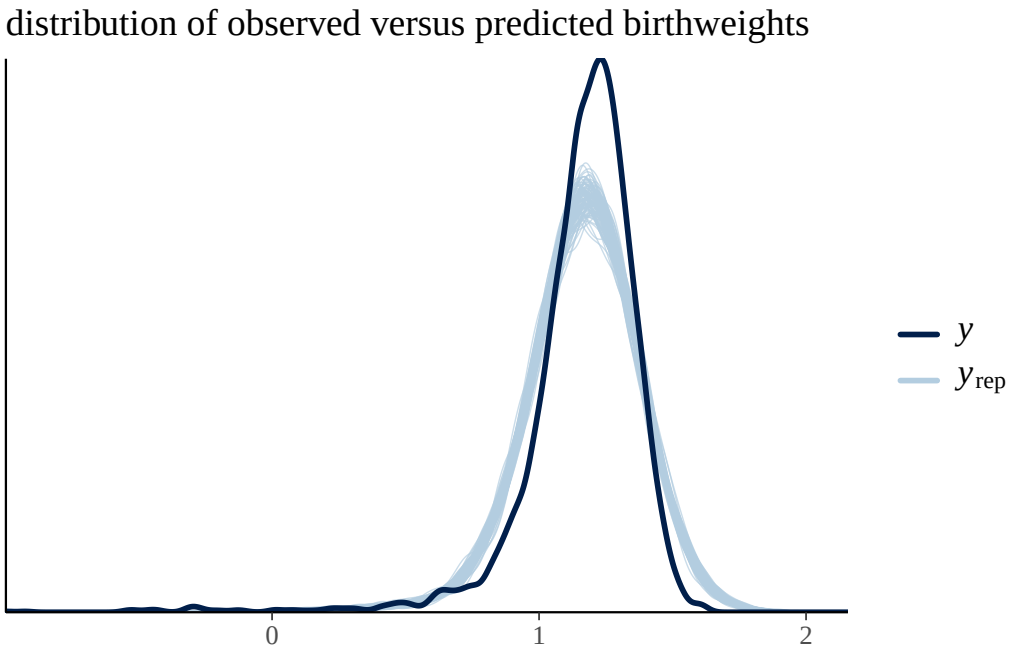
```
beta[2] 0.5990427 0.6554967  98.98279 1.0088166
beta[3] 0.1044230 0.1093843 613.22428 0.9978156
beta[4] 0.2064079 0.2182454 121.59685 1.0056875
sigma   0.1623019 0.1646189 320.75100 1.0104805
```

We see that our results are pretty similar, although some of our beta estimates are swapped. This is just because we may have inputted these variables in a different order.

PPCs

Now we've run two candidate models let's do some posterior predictive checks. The `bayesplot` package has a lot of inbuilt graphing functions to do this. For example, let's plot the distribution of our data (y) against 100 different datasets drawn from the posterior predictive distribution:

```
set.seed(1856)
y <- ds$log_weight
yrep1 <- extract(mod1)[["log_weight_rep"]]
yrep2 <- extract(mod2)[["log_weight_rep"]]
samp100 <- sample(nrow(yrep1), 100)
ppc_dens_overlay(y, yrep1[samp100, ])  +
  ggtitle("distribution of observed versus predicted birthweights")
```

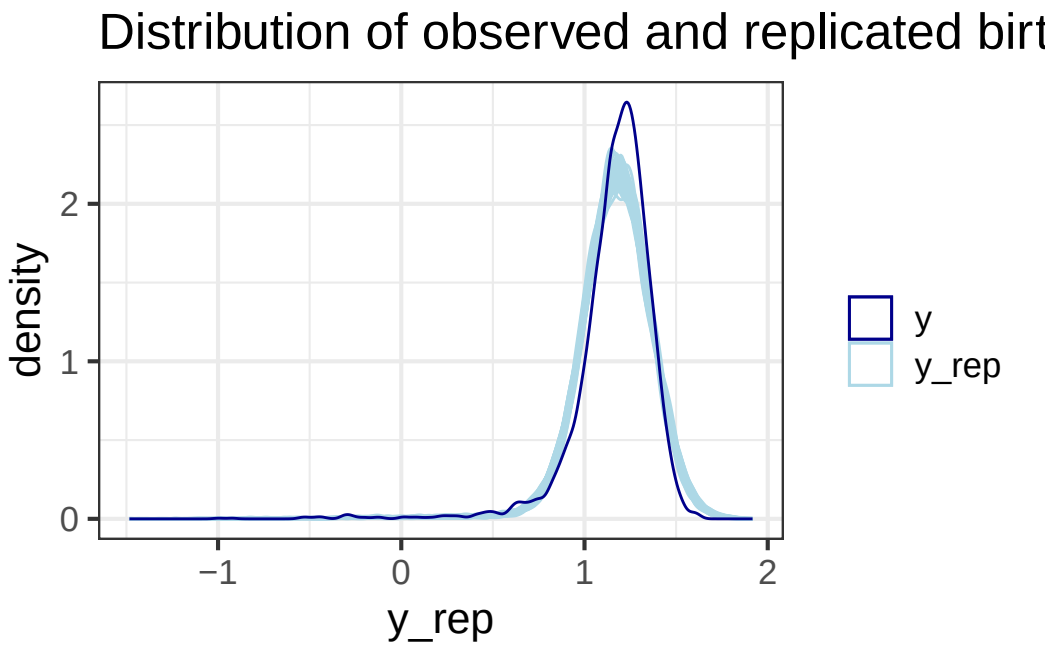


Question 6

Make a similar plot to the one above but for model 2, and **not** using the bayes plot in built function (i.e. do it yourself just with `geom_density`)

```
N <- nrow(ds)
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 for plotting
dr <- dr %>%
pivot_longer(-(i:log_weight_obs), names_to = "sim", values_to ="y_rep")
# Filter to include 100 draws and plot
dr %>%
filter(sim %in% samp100) %>%
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)
```

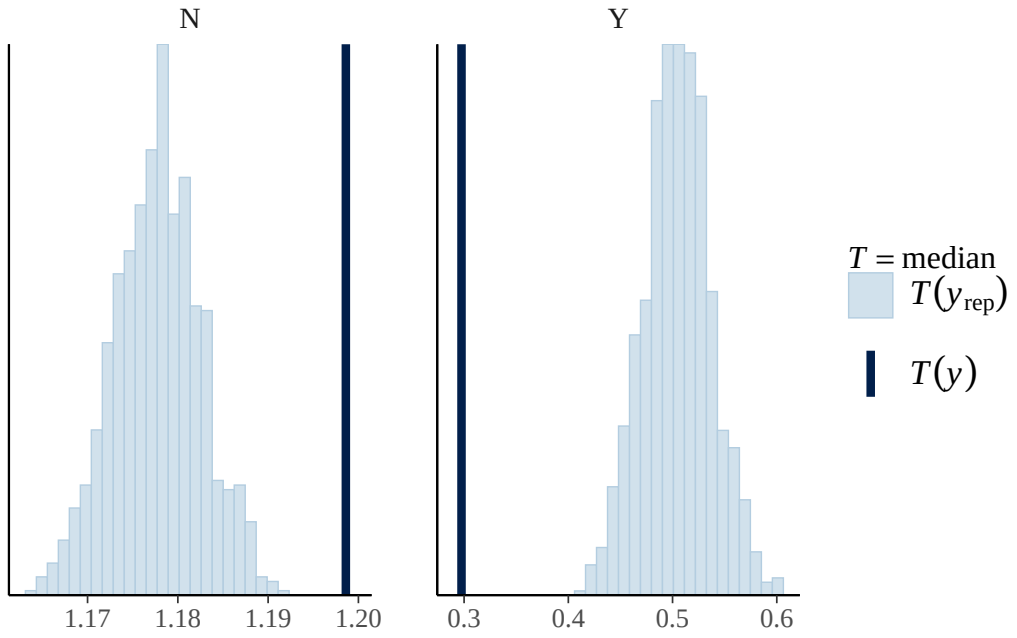


Test statistics

We can also look at some summary statistics in the PPD versus the data, again either using `bayesplot` – the function of interest is `ppc_stat` or `ppc_stat_grouped` – or just doing it ourselves using ggplot.

E.g. medians by prematurity for Model 1

```
ppc_stat_grouped(ds$log_weight, yrep1, group = ds$preterm, stat = 'median')
```

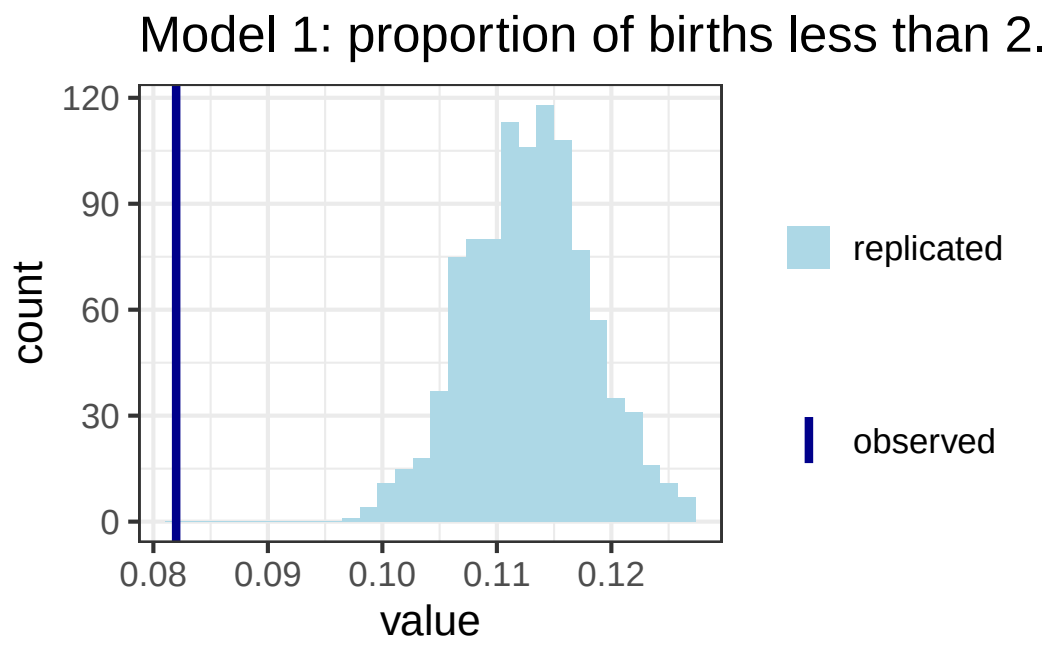


Question 7

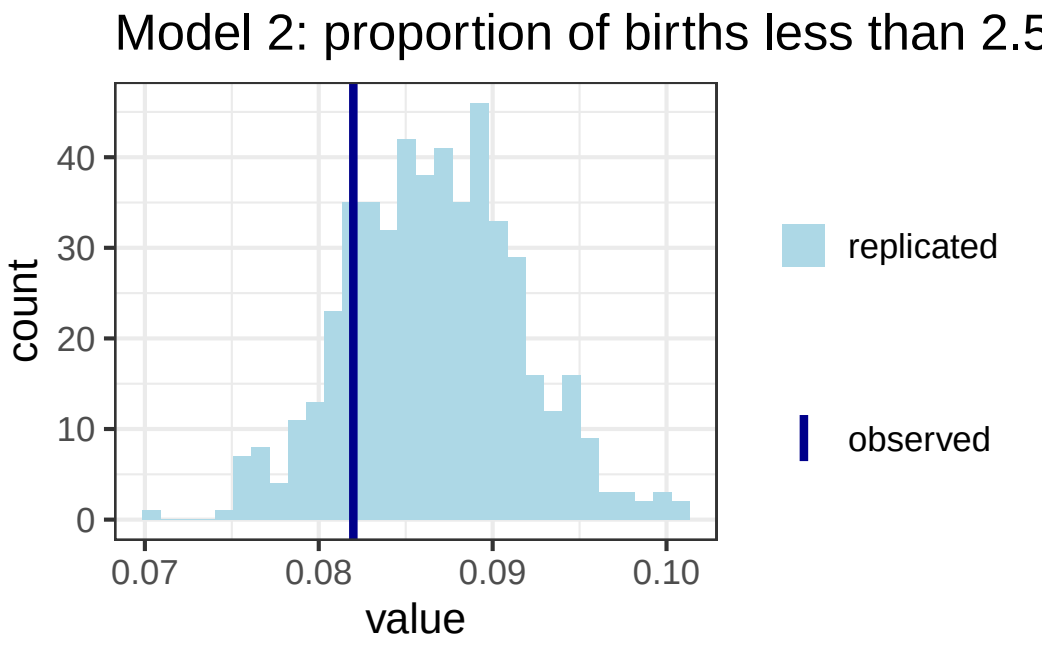
Use a test statistic of the proportion of births under 2.5kg. Calculate the test statistic for the data, and the posterior predictive samples for both models, and plot the comparison (one plot per model).

```
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"))
```




```
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"))
```



From the above diagrams, it is clear that Model 2 is better, since it can estimate the proportion of babies less than 2.5kg better.

LOO

Finally let's calculate the LOO elpd for each model and compare. The first step of this is to get the point-wise log likelihood estimates from each model:

```
loglik1 <- extract(mod1)[["log_lik"]]
loglik2 <- extract(mod2)[["log_lik"]]
```

And then we can use these in the `loo` function to get estimates for the elpd. Note the `save_psis = TRUE` argument saves the calculation for each simulated draw, which is needed for the LOO-PIT calculation below.

```
loo1 <- loo(loglik1, save_psis = TRUE)
loo2 <- loo(loglik2, save_psis = TRUE)
```

Look at the output:

```
loo1
```

```
Computed from 1000 by 3842 log-likelihood matrix

         Estimate    SE
elpd_loo   1377.0  72.4
p_loo         9.8   1.4
looic     -2754.1 144.8
------
Monte Carlo SE of elpd_loo is 0.1.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

```
loo2
```

```
Computed from 500 by 3842 log-likelihood matrix

         Estimate    SE
elpd_loo   1552.8  70.0
p_loo        14.8   2.3
looic     -3105.6 139.9
------
Monte Carlo SE of elpd_loo is 0.2.

All Pareto k estimates are good (k < 0.5).
See help('pareto-k-diagnostic') for details.
```

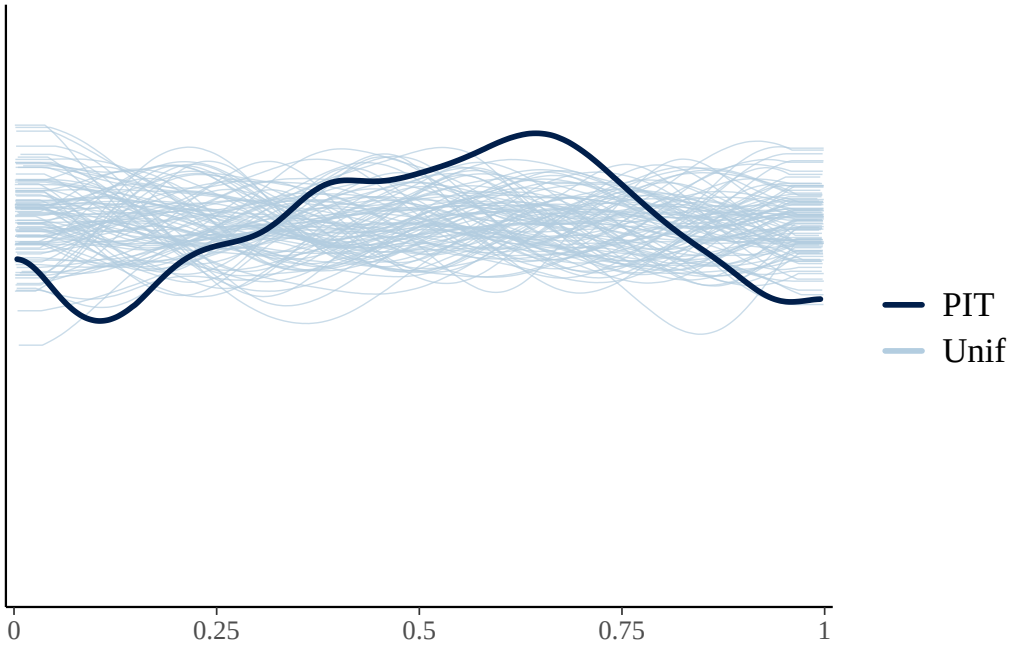
Comparing the two models tells us Model 2 is better:

```
loo_compare(loo1, loo2)
```

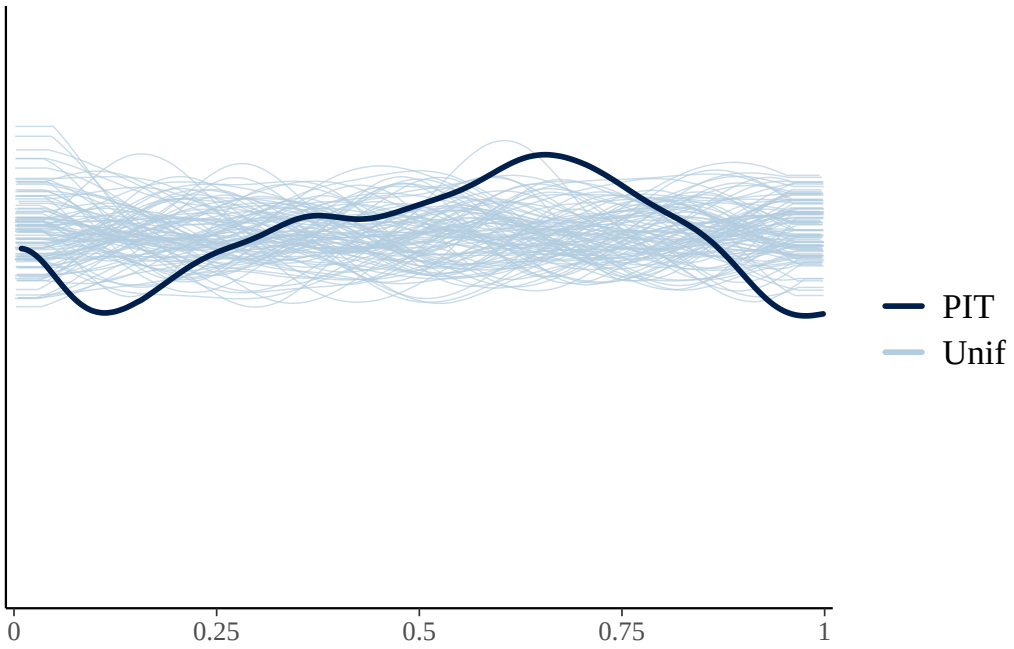
```
       elpd_diff se_diff
model2    0.0       0.0
model1 -175.8      36.2
```

We can also compare the LOO-PIT of each of the models to standard uniforms. The both do pretty well.

```
ppc_loo_pit_overlay(yrep = yrep1, y = y, lw = weights(loo1$psis_object))
```




```
ppc_loo_pit_overlay(yrep = yrep2, y = y, lw = weights(loo2$psis_object))
```



Bonus question (not required)

Create your own PIT histogram "from scratch" for Model 2.

Question 8

Based on the original dataset, choose one (or more) additional covariates to add to the linear regression model. Run the model in Stan, and compare with Model 2 above on at least 2 posterior predictive checks.

I will add the age of the mother to our model. This is the `mager` covariate. Earlier we plotted the age against the birthweight, and noticed some interesting trends in our EDA. Thus, it makes sense to include it now.

```
ds$log_weight <- log(ds$birthweight)
ds$log_gest_c <- (log(ds$gest) - mean(log(ds$gest)))/sd(log(ds$gest))
ds$preterm <- ifelse(ds$preterm == 'Y', 1, 0)

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

```
                  log_gest = ds$log_gest_c,
                  age = ds$mager,
                  preterm = ds$preterm)
```

Now fit the model

```
mod3 <- stan(data = stan_data,
             file = here("code/models/simple_weight_preterm_age.stan"),
             iter = 1000,
             seed = 243)
```

```
summary(mod3)$summary[c("beta[1]", "beta[2]", "beta[3]", "beta[4]", "beta[5]", "sigma"),]
```

```
              mean      se_mean           sd        2.5%         25%       50%
beta[1] 1.095952748 3.885746e-04 0.0130516426 1.070672560 1.087600788 1.0957195
beta[2] 0.102928400 7.931447e-05 0.0035548466 0.095981895 0.100552421 0.1028750
beta[3] 0.563105360 2.334634e-03 0.0604477373 0.446620651 0.521297857 0.5632074
beta[4] 0.002548269 1.335582e-05 0.0004446381 0.001675768 0.002250641 0.0025528
beta[5] 0.197784128 4.713633e-04 0.0123779193 0.174263896 0.189402917 0.1979139
sigma   0.160575773 5.255369e-05 0.0017929210 0.157129115 0.159345688 0.1605234
                75%       97.5%     n_eff     Rhat
beta[1] 1.104720366 1.122077959 1128.1881 1.003282
beta[2] 0.105194913 0.110150351 2008.8006 1.000766
beta[3] 0.603280114 0.682209281  670.3823 1.004962
beta[4] 0.002841812 0.003430914 1108.3385 1.003342
beta[5] 0.206327784 0.221324856  689.5785 1.004191
sigma   0.161770144 0.164051544 1163.9007 1.002435
```

Now, we will do some posterior predictive checks.

Let's calculate the LOO elpd for models 2 and the new model 3

```
loglik2 <- extract(mod2)[["log_lik"]]
loglik3 <- extract(mod3)[["log_lik"]]
```

use these to get the estimates for elpd.

```
loo2 <- loo(loglik2, save_psis = TRUE)
loo3 <- loo(loglik3, save_psis = TRUE)
```

And now we compare the two

```
loo_compare(loo2, loo3)
```

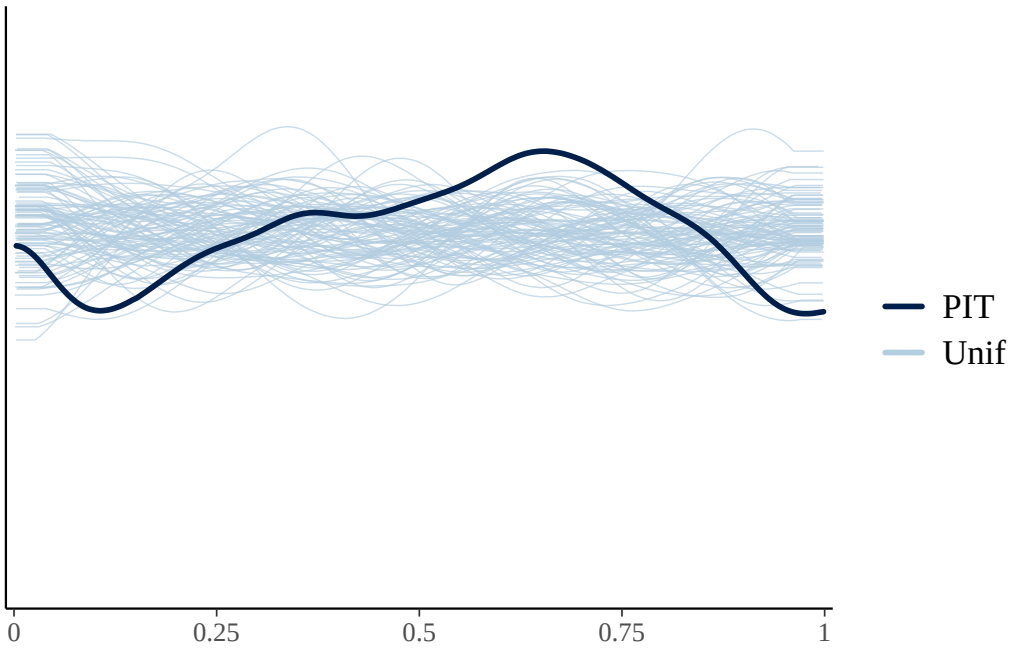
```
       elpd_diff se_diff
model2   0.0       0.0
model1 -15.3       5.7
```

We see that model 2 has an elpd difference of 15.3 units and our standard error difference is 5.7. This is around three standard errors, so this is a significant result. However, model1 has a negative value, so model1 is worse. Based on this, we can conclude our newer model (model 3) is better. Note also that the model names in the table don't correspond correctly, as model2 is model 3.

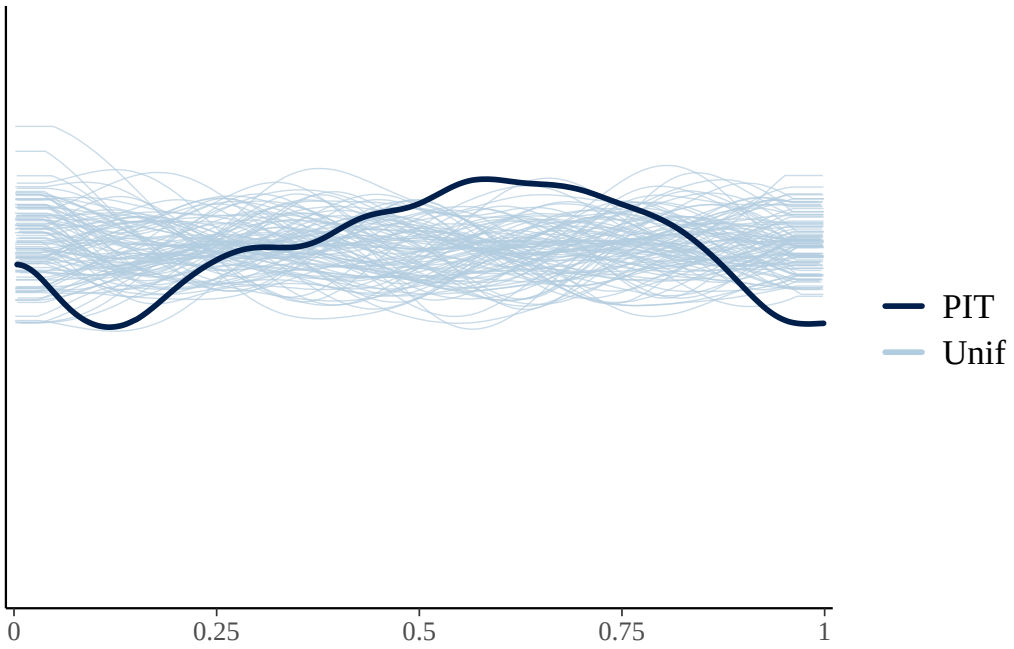
Let's also compare to standard uniforms, as we know theoretically LOO-PIT should be uniform(ish).

```
yrep3 <- extract(mod3)[["log_weight_rep"]]

ppc_loo_pit_overlay(yrep = yrep2, y = y, lw = weights(loo2$psis_object))
```




```
ppc_loo_pit_overlay(yrep = yrep3, y = y, lw = weights(loo3$psis_object))
```



Both models are somewhat uniform, and this is good enough by this diagnostic. Therefore, we could choose the model without maternal age, as this seems to be doing pretty good by this most recent check. However, by comparing the elpd's, model 3 was better compared to model 2. I would choose model3, trading accuracy for parsimony.