Group Work 06

```r
library(bayesrules) # R package for our textbook
library(tidyverse) # Collection of packages for tidying and plotting data
library(janitor) # Helper functions like tabyl
library(rstan) # for fitting models
library(rstanarm) # for fitting standard regression models
library(broom.mixed) # for tidy() function
library(bayesplot) # helpful plotting functions
library(tidybayes) # helpful for wrangling Bayesian model output
```

> **ℹ Note**
>
> I've given you most of the R code you need for today to help you get through the
> questions more quickly and focus on the concepts. Make sure you understand what all
> the code is doing, enough so that you can replicate it on your own in the future.
> In code chunks where you need to fill in the blank, remember to remove #| `eval: false`
> to get your document to render correctly

Throughout these problems, you'll be modeling the daily 3pm temperature in Perth, Australia,
based on the 9am temperature. We'll use the `weather_perth` data in the {bayesrules}
package

```r
data("weather_perth")
```

We'll use the following Bayesian regression model:

$$Y_i | \beta_0, \beta_1, \sigma \sim N(\mu_i, \sigma^2)$$

$$\mu_i = \beta_0 + \beta_1 X_i$$

$$\beta_{0c} \sim N(30, 7^2)$$

$$\beta_1 \sim N(2, 1^2)$$

$$\sigma \sim Exp(0.1)$$

1 Prior understanding

(a) Start by summarizing our prior knowledge of $\beta_{0c}, \beta_1, \sigma$. Fill in the blanks to plot the appropriate prior distributions, then provide a 1 sentence summary of each parameter.

```
p1 <- plot_normal(___, ___) +
        labs(x = expression(beta[0c]), y = "density")

p2 <- plot_normal(___, ___) +
        labs(x = expression(beta[1]), y = "density")

p3 <- plot_gamma(___, ___) +
        labs(x = expression(sigma), y = "density")

p1 + p2 + p3
```

(b) The prior models help us understand each individual parameter, $\beta_{0c}, \beta_1, \sigma$. To better understand how these work together in the model of 3pm temperatures, we can simulate a sample of prior plausible parameter sets using the `stan_glm()` function in the {rstanarm} package. Fill in the blanks in the code, then annotate each line of code with a comment about what it does.

```
# Simulate the prior model
temp_prior <- stan_glm(
  temp3pm ~ temp9am,  # your comment here
  data = weather_perth,
  family = gaussian,  # your comment here
  prior_intercept = normal(___, ___),  # your comment here
  prior = normal(___, ___),  # your comment here
  prior_aux = exponential(___),  # your comment here
  prior_PD = TRUE,  # your comment here
  chains = 4, iter = 5000*2, seed = 84735) # your comment here
```

(c) `temp_prior` should contain 20,000 plausible parameter sets. It's important to check our priors The following chunk simulates 200 prior plausible regression lines $\beta_0 + \beta_1 X$, 1 from each of 200 prior plausible parameter sets in temp_prior. Run it a few times and summarize your observations.

```
# 200 prior model lines
weather_perth %>%
  add_epred_draws(temp_prior, ndraws = 200) %>%
  ggplot(aes(x = temp9am, y = temp3pm)) +
    geom_line(aes(y = .epred, group = .draw), alpha = 0.15)
```

(d) The plot in part c illuminates prior plausible regression lines $\beta_0 + \beta_1 X$. To explore our prior understanding of potential variability from this line, σ, the following chunk simulates 4 prior plausible sets of temperature data, 1 from each of 4 prior plausible parameter sets in temp_prior. Run it a few times and summarize your observations.

```
# Plot 4 prior simulated datasets
weather_perth %>%
  add_predicted_draws(temp_prior, ndraws = 4) %>%
  ggplot(aes(x = temp9am, y = temp3pm)) +
    geom_point(aes(y = .prediction, group = .draw), size = 0.1) +
    facet_wrap(~ .draw)
```

(e) Explain the difference between the analysis in (c) and in (d)

2 Simulate the posterior

(a) Next, let's simulate the posterior. We don't have to start from scratch. Rather, we can update() our prior simulation using prior_PD = FALSE.

```
# Simulate the posterior
```

(b) To ensure that we can "trust" the MCMC posterior simulation results, create trace plots and density plots of the multiple chains.

```
# your code here
```

(c) Just as we did for the prior, we can better understand our posterior model by simulating 50 posterior plausible regression lines $\beta_0 + \beta_1 X$. Run this chunk a few times. Summarize your observations as well as how our understanding evolved from the prior to the posterior.

```
## 50 simulated posterior model lines

___ %>%
  ___(___, ndraws = 50) %>%
  ggplot(aes(x = ___, y = ___)) +
    geom_line(aes(y = ___, group = ___), alpha = 0.15) +
    geom_point(data = weather_perth, size = 0.05)
```

(d) Check out a posterior summary table. Do we have ample evidence of a positive association between 3pm and 9am temperatures, i.e. that the warmer it is in the morning, the warmer it tends to be in the afternoon? Explain.

```
# Get posterior summaries of the "fixed" parameters (betas)
# and "auxiliary" parameter (sigma)
tidy(temp_posterior, effects = c("fixed", "aux"),
     conf.int = TRUE, conf.level = 0.80)
```

3 Posterior prediction

(a) Suppose it's 20 degrees at 9am. Simulate, plot, and describe a posterior predictive model for the 3pm temperature. HINTS: Predict a 3pm temperature from each of the 20,000 parameter sets. To work with (Intercept), put a backtick at the start and end.

```
# Store the MCMC chains as a data frame
 temp_chains <- as.data.frame(temp_posterior)

# Simulate the posterior predictive model
 set.seed(84735)
 predict_20 <- ___ %>%
   ___(y_prediction = ___)

# Plot the posterior predictive model
 ggplot(predict_20, aes(x = ___)) +
   geom_density()
```

(b) It's important to understand where the predictions come from. BUT there's also a short-cut. Use the posterior_predict() function to simulate the posterior predictive model of 3pm temperature when it's 20 degrees at 9am. Further, use posterior_interval() and mcmc_dens() to summarize the posterior (use a 90% level). Examine the results and confirm that they're similar to your "from scratch" simulation.

```
# Simulate a set of predictions
set.seed(84735)
shortcut_prediction <- posterior_predict(
    temp_posterior, newdata = data.frame(temp9am = 20))

# Construct a 90% posterior credible interval
posterior_interval(shortcut_prediction, prob = ___)

# Plot the approximate predictive model
```

```
mcmc_areas(shortcut_prediction, prob = ___) +
  scale_x_continuous(breaks = seq(15, 35, by = 5)) +
  xlab("3pm temperature")
```

4 Posterior predictive check

In the question above, we considered posterior predictions for a single (new) data point. We can also use posterior predictions *on the entire original dataset* to evaluate how well our model is performing.

(a) The code below (1) pulls out the first set of $(\beta_0, \beta_1, \sigma)$ from our MCMC sample and then (2) simulates a prediction $\hat{y}$ based on those estimates, for every point in the original dataset. Fill in the blanks and then take a peek at **one_simulation** to make sure you understand what it contains.

```
first_set <- head(temp_chains, 1)

beta_0 <- first_set$`(Intercept)`
beta_1 <- first_set$temp9am
sigma  <- first_set$sigma

set.seed(84735)
one_simulation <- weather_perth %>%
  mutate(mu = ___ + ___ * ___,
         simulated_temp3am = rnorm(500, mean = mu, sd = ___)) %>%
  select(temp9am, temp3pm, simulated_temp3pm)
```

Next, run this chunk of code to create a density plot of the predictions compared to the observed.

```
ggplot(one_simulation, aes(x = temp9am)) +
  geom_density(color = "lightblue") +
  geom_density(aes(x = rides), color = "darkblue")
```

(b) It's *really important* to have a solid understanding of what a posterior predictive check is doing under the hood. But one of the benefits of using {rstanarm} is that, since all of the models have a consistent structure, we can make use of shortcut functions that exist in other packages. Use **pp_check** below to examine 50 simulated samples. Write a 1-sentence summary of your findings.

```
# Examine 50 of the 20000 simulated samples
pp_check(temp_model, nreps = 50)
```