Lab 7

Your Name

Hierarchical linear regression

In this lab we will test our understanding of random effects (mixed-effects) modeling.

```
#---------------------------------------
#---------------------------------------
# Import/Load the rstan library:
library(tidyverse)
library(rstan)
library(loo)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
#---------------------------------------
#---------------------------------------
## Simulating the data
```

Simulating the data

Our first task is to simulate data following the assumptions of random-effects modeling. First, we will assume a true model in which groups have an equal slope, but vary in their intercept:

$$y_i = (\bar{\alpha} + \eta_{j,[i]}) + \beta x_i$$

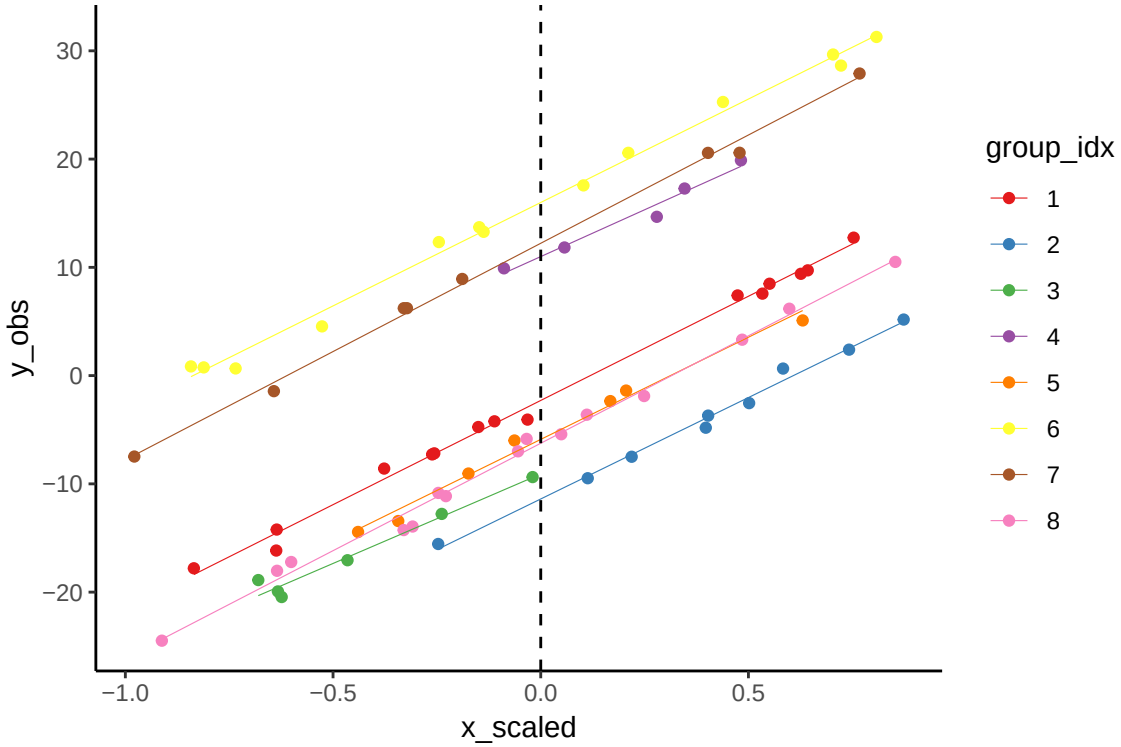
Task 1 (25 points)

Complete the following:

1. Add comments to this code-chunk to convey your understanding (2 points).

```
set.seed(7)

n_group = 8

n_samp_per_group = ceiling(runif(n_group, 4, 15))

group_idx = rep(c(1:n_group), times = n_samp_per_group)

n_sample = sum(n_samp_per_group)

alpha_mean = rnorm(1, 0, 10)
alpha_sigma = abs(rnorm(1, 0, 10))
alpha_eta_vec = rnorm(n_group, 0, alpha_sigma)

beta = rnorm(1, 0, 10)
sigma_resid = abs(rnorm(1, 0, 2))

x_raw = runif(n_sample, 0, 3500)
```

```r
x_mean = mean(x_raw)
x_sd = sd(x_raw)
x_scaled = (x_raw - x_mean) / (2 * x_sd)

epsilon = rnorm(n_sample, 0, sigma_resid)


y_hat = NULL
for(i in 1:n_sample){

  y_hat[i] = (alpha_mean + alpha_eta_vec[group_idx[i]]) + x_scaled[i] * beta

}
y_obs = y_hat + epsilon


data_df =
  data.frame(y_obs, x_scaled, group_idx) %>%
  mutate(group_idx = factor(group_idx))


ggplot(data_df, aes(x = x_scaled, y = y_obs, color = group_idx)) +
  geom_point(shape = 19) +
  geom_smooth(method = "lm", se = FALSE, size = 0.2) +
  scale_color_brewer(palette = "Set1") +
  geom_vline(xintercept = 0, linetype = 2) +
  theme_classic()
```



Fitting the Stan models

2. Use the supplied Stan code to fit the random effects model to your generated data. **Hint**: Remember to monitor your `log_lik` variable so that you can complete sub-task 5, below (5 points).
3. Construct at least two visualizations to validate that your model was able to adequately recover the true parameters. Note: A table can be considered a visualization, because it organizes the quantitative information in a visually accessible manner (5 points).
4. Construct a Stan code file that allows for both random intercepts and random slopes. Fit this new, more complex model to your same data set (8 points).
5. Calculate the LOO-IC for each of your two models. Comment on which model is more parsimonious, given the data at hand (5 points).

Task 2 (15 points)

1. Subset your data to include only one of the groups.
2. For that specific group, create a scatterplot.

(For the following, use your most parsimonious model)

3. On the scatterplot, overlay the median model-fit (i.e., the median estimated line).
4. Bootstrap from the posterior to overlay many more possible model-fits. Be sure that for each of these model-fits, you are drawing *joint* parameter sets from your MCMC samples. **Hint**: To get a joint draw, first specify which sample you will take from your MCMC chains. For instance:

```
#...
  temp_idx = sample(c(1:n_total_mcmc_samples), 1)
  temp_alpha_mean = model_out$alpha_mean[temp_idx]
#...
```

5. Make sure it is easy to visually distinguish between the median model-fit and the other possible model-fits. (You might want to plot the "other" fits before overlaying the median fit).