

Lab 8

Bingying Liu

3/23/2020

Exercise 1: Simple EDA: for each school, calculate the sample average of the course scores and plot the distribution of these averages in a histogram. What does this plot reveal?

```
data(Gcsemv, package = "mlmRev")

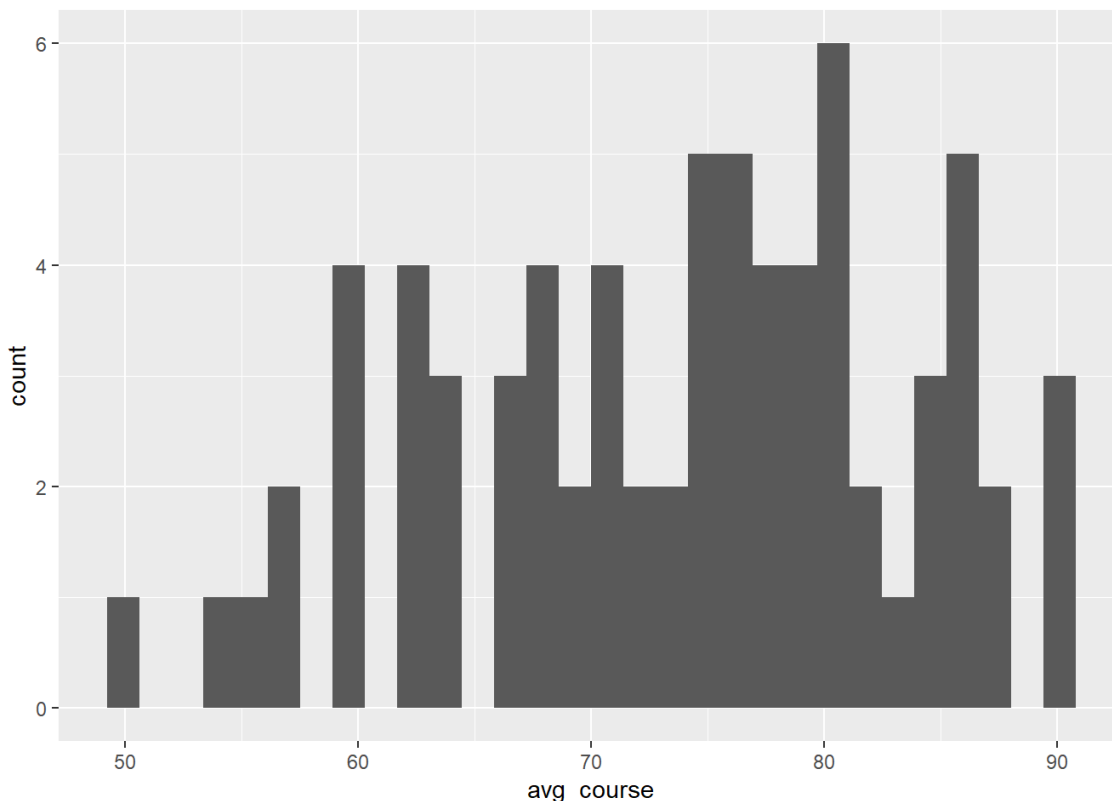
# Make Male the reference category and rename variable
Gcsemv$female <- relevel(Gcsemv$gender, "M")

# Use only total score on coursework paper
GCSE <- subset(x = Gcsemv,
               select = c(school, student, female, course))

# Count unique schools and students
m <- length(unique(GCSE$school))
N <- nrow(GCSE)
```

```
GCSE %>%
  dplyr::group_by(school) %>%
  na.omit()%>%
  dplyr::summarise(avg_course = mean(course)) %>%
  dplyr::ungroup() %>%
  ggplot(aes(x = avg_course))+
  geom_histogram()
```

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



This plot reveals that the school average of course falls within the range 50 to 90 with a mean about 75. Distribution of the school average is approximately normal.

```
# Pooled and unpooled models
pooled <- stan_glm(course ~ 1 + female, data = GCSE, refresh = 0)
unpooled <- stan_glm(course ~ -1 + school + female, data=GCSE, refresh = 0)
```

Model1: Variance components model

```
# Model1: Variance components model
mod1 <- stan_lmer(formula = course ~ 1 + (1 | school),
                  data = GCSE,
                  seed = 349,
                  refresh = 0)

prior_summary(object = mod1)
```

```
## Priors for model 'mod1'
## -----
## Intercept (after predictors centered)
##   Specified prior:
##     ~ normal(location = 0, scale = 10)
##   Adjusted prior:
##     ~ normal(location = 0, scale = 163)
##
## Auxiliary (sigma)
##   Specified prior:
##     ~ exponential(rate = 1)
##   Adjusted prior:
##     ~ exponential(rate = 0.061)
##
## Covariance
## ~ decov(reg. = 1, conc. = 1, shape = 1, scale = 1)
## -----
## See help('prior_summary.stanreg') for more details
```

```
# observed standard deviation of the course variable
sd(GCSE$course, na.rm = T)
```

```
## [1] 16.32096
```

```
print(mod1, digits = 3)
```

```
## stan_lmer
## family:      gaussian [identity]
## formula:     course ~ 1 + (1 | school)
## observations: 1725
## -----
##              Median MAD_SD
## (Intercept) 73.749  1.137
##
## Auxiliary parameter(s):
##      Median MAD_SD
## sigma 13.823  0.230
##
## Error terms:
## Groups   Name      Std.Dev.
## school   (Intercept) 8.917
## Residual              13.824
## Num. levels: school 73
##
## -----
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
summary(mod1,
  pars = c("(Intercept)", "sigma", "Sigma[school:(Intercept),(Intercept)]"),
  probs = c(0.025, 0.975),
  digits = 3)
```

```
##
## Model Info:
## function:      stan_lmer
## family:        gaussian [identity]
## formula:       course ~ 1 + (1 | school)
## algorithm:     sampling
## sample:        4000 (posterior sample size)
## priors:        see help('prior_summary')
## observations:  1725
## groups:        school (73)
##
## Estimates:
##              mean      sd      2.5%    97.5%
## (Intercept)    73.761   1.167   71.468   75.985
## sigma          13.824   0.236   13.362   14.292
## Sigma[school:(Intercept),(Intercept)] 79.514  15.553  54.263 114.027
##
## MCMC diagnostics
##              mcse  Rhat  n_eff
## (Intercept)    0.043 1.005   745
## sigma          0.003 1.000 5793
## Sigma[school:(Intercept),(Intercept)] 0.490 1.004 1006
##
## For each parameter, mcse is Monte Carlo standard error, 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).
```

Exercise 2: Report on the posterior estimates of μ_θ , σ and τ^2 .

$E(\mu_\theta|Y) = 73.761$, $E(\sigma|Y) = 13.824$, $E(\tau^2|Y) = 79.514$.

```

## Posterior draws
mod1_sims <- as.matrix(mod1)
par_names <- colnames(mod1_sims)

# obtain draws for mu_theta
mu_theta_sims <- as.matrix(mod1, pars = "(Intercept)")

# obtain draws for each school's contribution to intercept
theta_sims <- as.matrix(mod1,
  regex_pars = "b\\[\\(Intercept\\) school\\:"])

# to finish: obtain draws for sigma and tau^2
sig_sims <- as.matrix(mod1,
  pars = "sigma")
tau2_sims <- as.matrix(mod1,
  pars = "Sigma[school:(Intercept),(Intercept)]")

# 73 school's varying intercept
int_sims <- as.numeric(mu_theta_sims) + theta_sims

# posterior mean
int_mean <- apply(int_sims, MARGIN = 2, FUN = mean)

# credible interval
int_ci <- apply(int_sims, MARGIN = 2, FUN = quantile, probs = c(0.025, 0.975))
int_ci <- data.frame(t(int_ci))

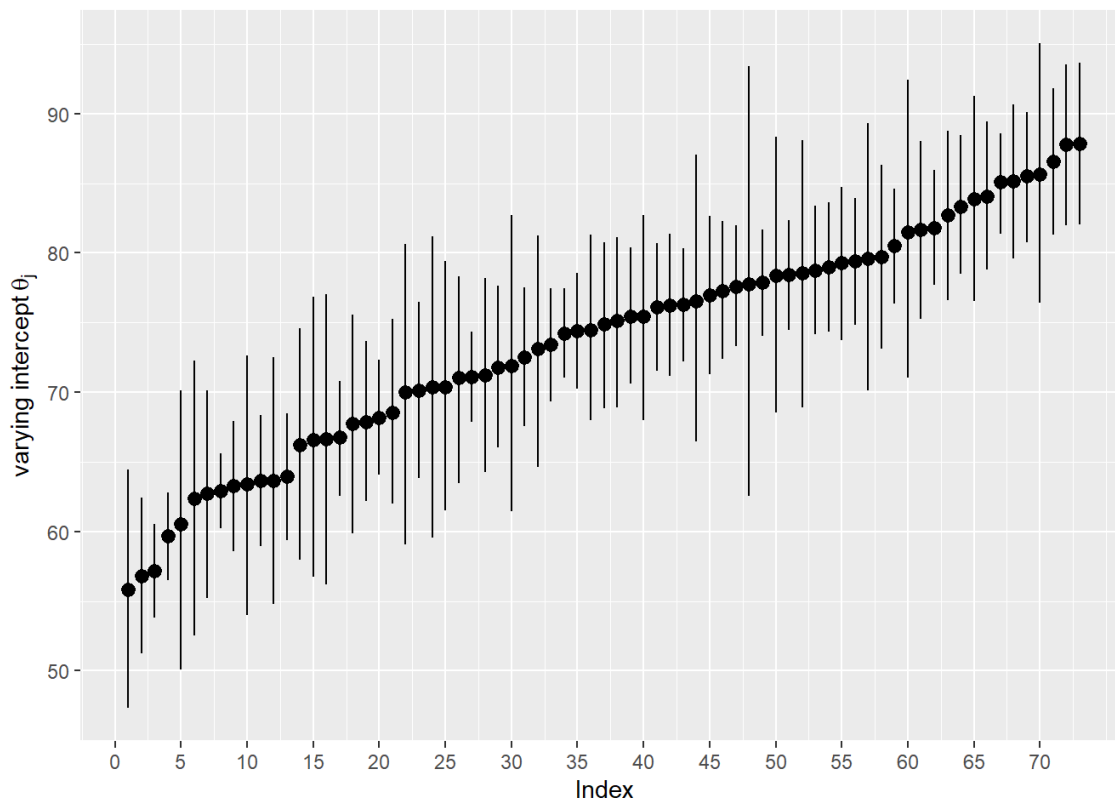
# combine into a single df
int_df <- data.frame(int_mean, int_ci)
names(int_df) <- c("post_mean", "Q2.5", "Q97.5")

# sort DF according to posterior mean
int_df <- int_df[order(int_df$post_mean),]

# create variable "index" to represent order
int_df <- int_df %>% mutate(index = row_number())

# plot posterior means of school-varying intercepts, along with 95 CIs
ggplot(data = int_df, aes(x = index, y = post_mean))+
  geom_pointrange(aes(ymin = Q2.5, ymax = Q97.5))+
  scale_x_continuous("Index", breaks = seq(0,m, 5)) +
  scale_y_continuous(expression(paste("varying intercept ", theta[j])))

```



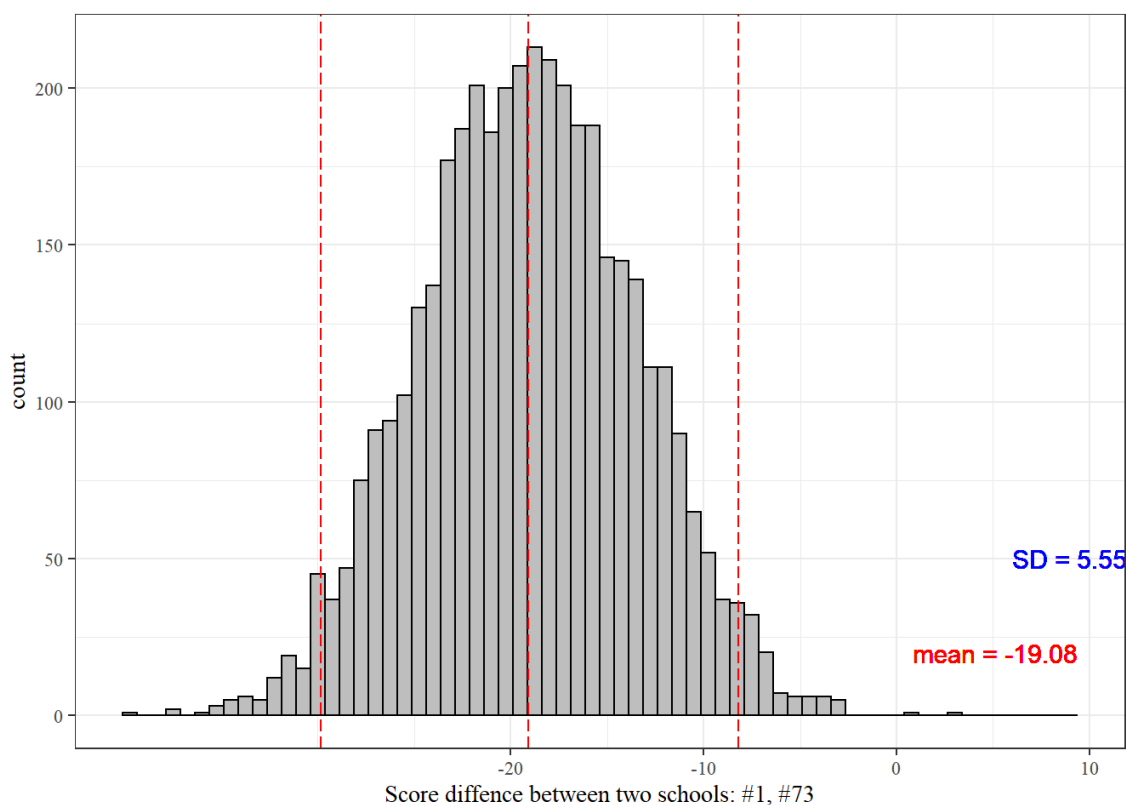
Exercise 3: Choose two schools and report on their difference in average scores with descriptive statistics a histogram and interpretation

```
# the difference between the two school averages (school #1 and #73)
school_diff = int_sims[,1] - int_sims[,73]

# investigate differences of two distributions
mean = mean(school_diff)
sd = sd(school_diff)
quantile = quantile(school_diff, probs = c(0.025, 0.50, 0.975))
names(quantile) = c("Q2.5", "Q50", "Q97.5")
diff_df = data.frame(mean, sd, quantile)
round(diff_df, 2)
```

```
##      mean  sd quantile
## Q2.5 -19.08 5.55  -29.82
## Q50  -19.08 5.55  -19.07
## Q97.5 -19.08 5.55   -8.21
```

```
# histogram of the differences
ggplot(data = data.frame(school_diff),
      aes(x = school_diff)) +
  geom_histogram(color = "black",
    fill = "gray",
    binwidth = 0.75) +
  scale_x_continuous("Score difference between two schools: #1, #73",
    breaks = seq(from = -20,
      to = 20,
      by = 10)) +
  geom_vline(xintercept = c(mean(school_diff),
    quantile(school_diff,
      probs = c(0.025, 0.975))),
    colour = "red",
    linetype = "longdash") +
  geom_text(aes(5.11, 20, label = "mean = -19.08"),
    color = "red",
    size = 4) +
  geom_text(aes(9, 50, label = "SD = 5.55"),
    color = "blue",
    size = 4) +
  theme_bw( base_family = "serif")
```



Interpretation: The expected

difference comes to -19.08 with a standard deviation of 5.55 and a wide range of uncertainty. The 95% credible interval is $[-29.82, -8.21]$, so we are 95% certain that the true value of the difference between the two schools lies within the range, given the data.

```
# we also can get the proportion of the time that school 20920 has a lower mean than school 68255:
prop.table(table(int_sims[,1]<int_sims[,73]))
```

```
##
## FALSE TRUE
## 0.0005 0.9995
```

This means that the posterior probability that school 20920 is worse than school 68225 is 99.9%.

Model 2: Varying intercept with a single individual-level predictor

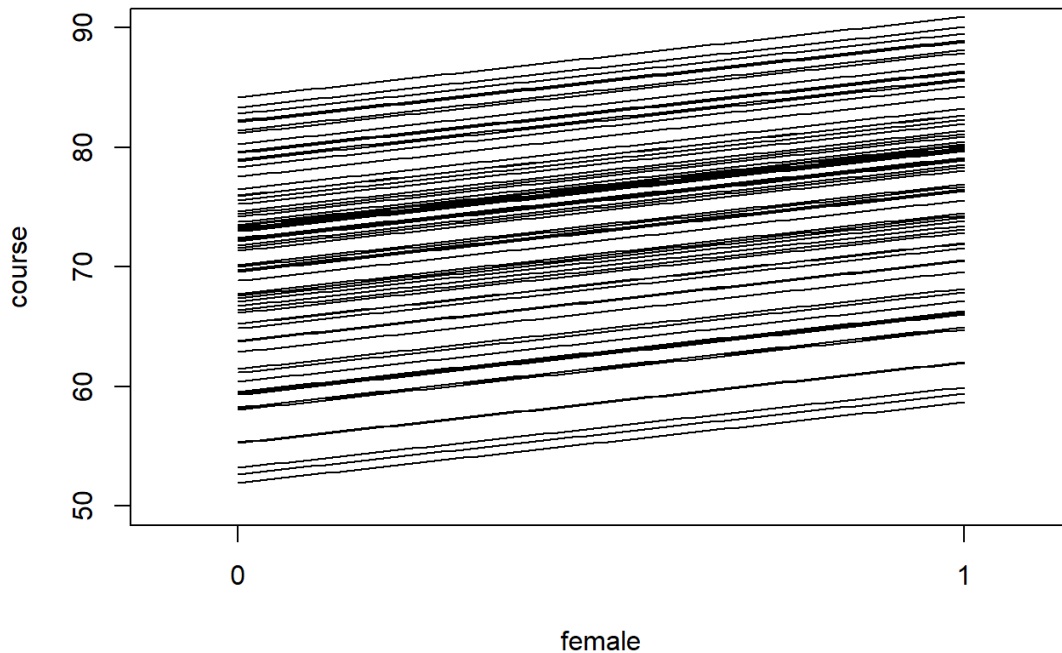
```

mod2 <- stan_lmer(formula = course ~ 1 + female + (1 | school),
  data = GCSE,
  prior = normal(location = 0,
    scale = 100,
    autoscale = FALSE),
  prior_intercept = normal(location = 0,
    scale = 100,
    autoscale = F),

  seed = 349,
  refresh = 0)

# plot varying intercepts
mod2.sims <- as.matrix(mod2)
group_int <- mean(mod2.sims[,1]) #mu_theta
mp <- mean(mod2.sims[,2]) #female F
bp <- apply(mod2.sims[, 3:75], 2, mean)
xvals <- seq(0,1,.01)
plot(x = xvals, y = rep(0, length(xvals)),
  ylim = c(50, 90), xlim = c(-0.1,1.1), xaxt = "n", xlab = "female", ylab = "course")
axis(side = 1, at = c(0,1))
for (bi in bp){
  lines(xvals, (group_int + bi)+xvals*mp)
}

```



Exercise 4: What are the posterior means and credible intervals of μ_θ , β , σ and τ^2 ?

```

par_names2 = colnames(mod2.sims)
head(par_names2)

```

```

## [1] "(Intercept)"          "femaleF"
## [3] "b[(Intercept) school:20920]" "b[(Intercept) school:22520]"
## [5] "b[(Intercept) school:22710]" "b[(Intercept) school:22738]"

```

```

tail(par_names2)

```

```
## [1] "b[(Intercept) school:76631]"
## [2] "b[(Intercept) school:77207]"
## [3] "b[(Intercept) school:84707]"
## [4] "b[(Intercept) school:84772]"
## [5] "sigma"
## [6] "Sigma[school:(Intercept),(Intercept)]"
```

```
summary(mod2,
  pars = c("(Intercept)", "femaleF", "sigma", "Sigma[school:(Intercept),(Intercept)]"),
  probs = c(0.025, 0.975),
  digits = 3)
```

```
##
## Model Info:
## function:      stan_lmer
## family:        gaussian [identity]
## formula:       course ~ 1 + female + (1 | school)
## algorithm:     sampling
## sample:        4000 (posterior sample size)
## priors:        see help('prior_summary')
## observations:  1725
## groups:        school (73)
##
## Estimates:
##               mean      sd    2.5%    97.5%
## (Intercept)    69.710   1.233   67.271   72.059
## femaleF         6.733   0.680    5.393    8.052
## sigma          13.424   0.234   12.964   13.890
## Sigma[school:(Intercept),(Intercept)] 81.289  16.558   54.788  121.203
##
## MCMC diagnostics
##               mcse  Rhat  n_eff
## (Intercept)    0.052 1.009   555
## femaleF        0.010 1.001  4351
## sigma          0.004 1.000  3861
## Sigma[school:(Intercept),(Intercept)] 0.715 1.005   536
##
## For each parameter, mcse is Monte Carlo standard error, 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).
```

$$E[\mu_{\theta}|Y] = 69.710, 95\% \text{ CI: } [69.271, 72.059]$$

$$E[\beta|Y] = 6.733, 95\% \text{ CI: } [5.393, 8.052]$$

$$E[\sigma|Y] = 13.424, 95\% \text{ CI: } [12.964, 13.890]$$

$$E[\tau^2|Y] = 81.289, 95\% \text{ CI: } [54.788, 121.203]$$

Model 3: Allowing for varying slopes across schools


```

mod3 <- stan_lmer(formula = course~ 1+ female + (1 + female | school),
                  data = GCSE,
                  seed = 349,
                  refresh = 0)
mod3_sims <- as.matrix(mod3)

# obtain draws for mu_theta
mu_theta_sims <- as.matrix(mod3, pars = "(Intercept)")

fem_sims <- as.matrix(mod3, pars = "femaleF")
# obtain draws for each school's contribution to intercept
theta_sims <- as.matrix(mod3,
                        regex_pars = "b\\[\\(Intercept\\) school\\:":)
beta_sims <- as.matrix(mod3,
                      regex_pars = "b\\[femaleF school\\:":)

int_sims <- as.numeric(mu_theta_sims) + theta_sims
slope_sims <- as.numeric(fem_sims) + beta_sims

# posterior mean
slope_mean <- apply(slope_sims, MARGIN = 2, FUN = mean)

# credible interval
slope_ci <- apply(slope_sims, MARGIN = 2, FUN = quantile, probs = c(0.025, 0.975))
slope_ci <- data.frame(t(slope_ci))

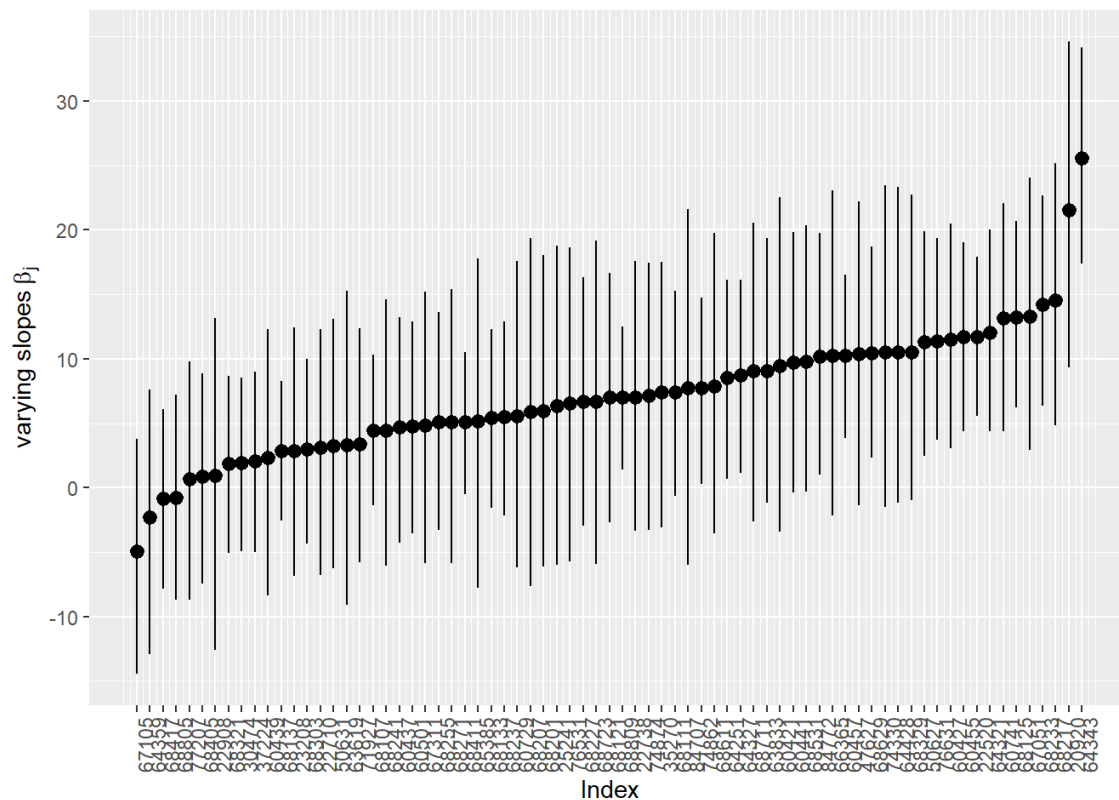
# combine into a single df
slope_df <- data.frame(slope_mean, slope_ci, levels(GCSE$school))
names(slope_df) <- c("post_mean", "Q2.5", "Q97.5", "school")

# sort DF according to posterior mean
slope_df <- slope_df[order(slope_df$post_mean),]

# create variable "index" to represent order
slope_df <- slope_df %>% mutate(index = row_number())

# plot posterior means of school-varying slopes, along with 95% CIs
ggplot(data = slope_df, aes(x = index, y = post_mean))+
  geom_pointrange(aes(ymin = Q2.5, ymax = Q97.5))+
  scale_x_continuous("Index", breaks = seq(1,m, 1),
                    labels = slope_df$school) +
  scale_y_continuous(expression(paste("varying slopes ", beta[j])))+
  theme(axis.text.x = element_text(angle = 90))

```



Model Comparison

```
loo1 <- loo(mod1)
loo2 <- loo(mod2)
loo3 <- loo(mod3)
loo_compare(loo1, loo2, loo3)
```

```
##      elpd_diff se_diff
## mod3    0.0      0.0
## mod2 -29.6      9.9
## mod1 -78.5     15.1
```

```

pooled.sim = as.matrix(pooled)
unpooled.sim = as.matrix(unpooled)
m1.sim = as.matrix(mod1)
m2.sim = as.matrix(mod2)
m3.sim = as.matrix(mod3)
schools = unique(GCSE$school)

alpha2 = mean(m2.sim[,1])
alpha3 = mean(m3.sim[,1])

partial.fem2 = mean(m2.sim[,2])
partial.fem3 = mean(m3.sim[,2])
unpooled.fem = mean(unpooled.sim[,74])

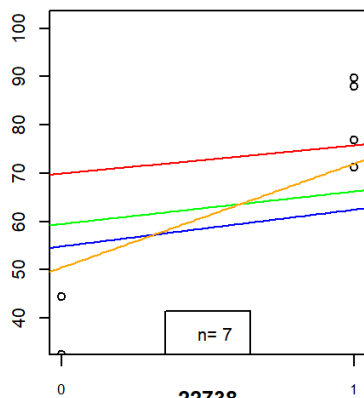
par(mfrow = c(2,3), mar = c(1,2,2,1))
for (i in 1:18){
  temp = GCSE %>% filter(school == schools[i]) %>%
  na.omit()
  y = temp$course
  x = as.numeric(temp$female) - 1
  plot(x + rnorm(length(x))*0.001, y, ylim = c(35,101), xlab = "female", main = schools[i], xaxt = "n", ylab = "course" )
  axis(1,c(0,1), cex.axis = 0.8)

  # no pooling
  b = mean(unpooled.sim[,i])

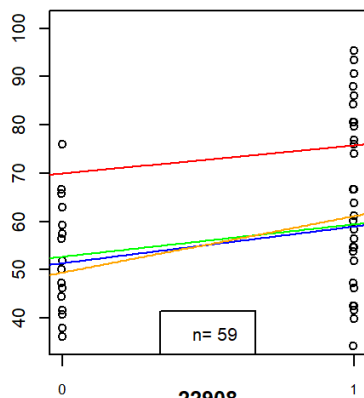
  # plot lines and data
  xvals = seq(-0.1,1.1,0.01)
  lines(xvals, xvals*mean(pooled.sim[,2]) + mean(pooled.sim[,1]), col = "red") #pooled
  lines(xvals, xvals * unpooled.fem + b, col = "blue") #unpooled
  lines(xvals, xvals * partial.fem2 + (alpha2 + mean(m2.sim[,i+2])), col = "green") # varying int
  lines(xvals, xvals*(partial.fem3 + mean(m3.sim[, 2 + i*2])) + (alpha3 + mean(m3.sim[, 1 + i*2])), col = "orange") # varyin
g int and slope
  legend("bottom", legend = paste("n=", length(y), " "))
}

```

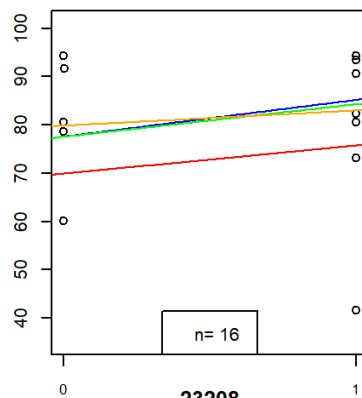
20920



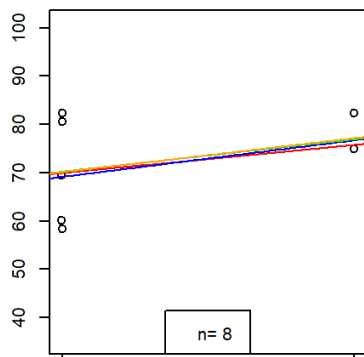
22520



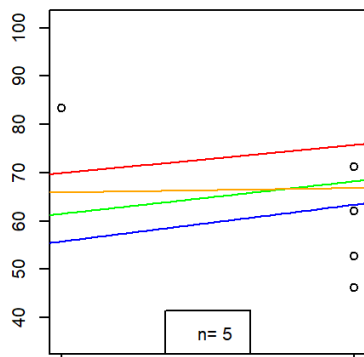
22710



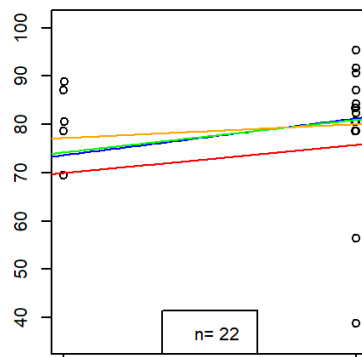
22738



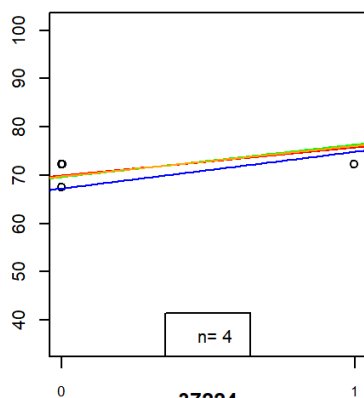
22908



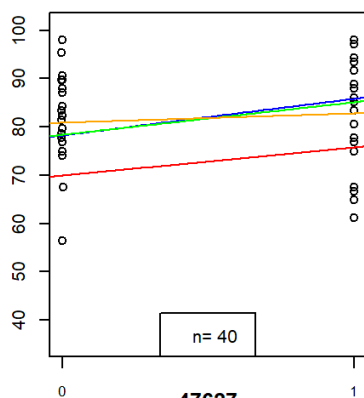
23208



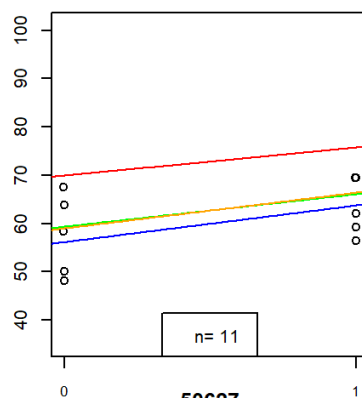
25241



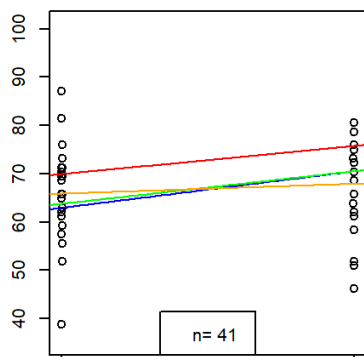
30474



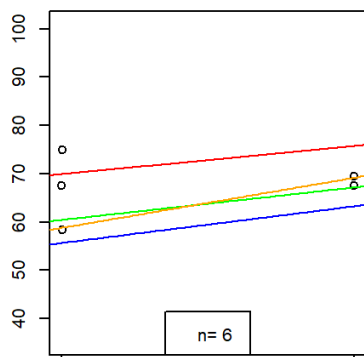
35270



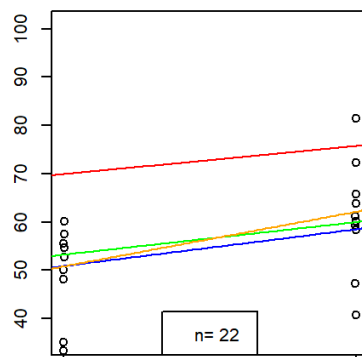
37224



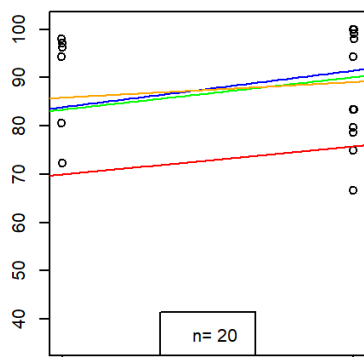
47627



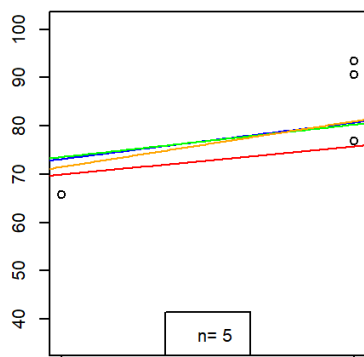
50627



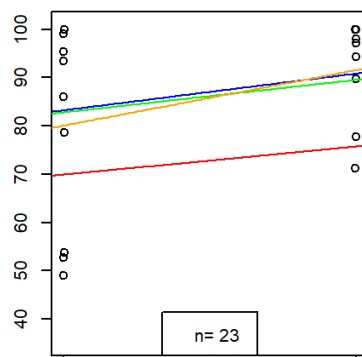
50631

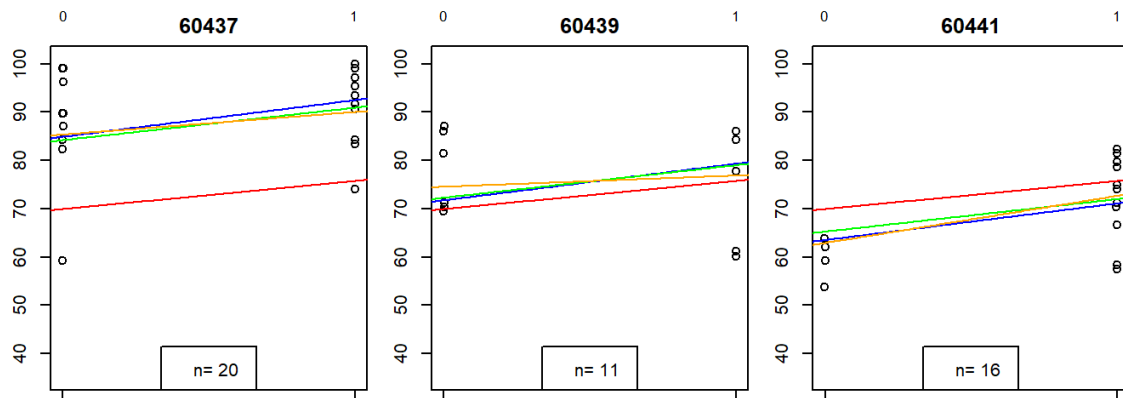


60421



60427





Exercise 5: Compare and contrast the regression lines estimated using these different methods. Also, based on the output of `loo_compare`, which model would you recommend?

On the graphs, we can see that pooled model performs significantly worse than unpooled and partial-pooling models, which are unpooled, mod2 and mod3. Since pooled ignore differences between groups and sometimes it's underfitting the group data. I would recommend the models with varying intercept and varying slope (mod3). Based on results in `loo_compare`, mod3 has the highest `elpd_diff`. And `elpd_diff` is the difference in `elpd_loo` for 2 models (in this case each `elpd_loo` is compared with the model with highest `elpd_loo`, which is mod3). `elpd_loo` measures the bayesian Loo estimate of expected log pointwise predictive density. On the graph, we can see that the orange is neither overfitting nor underfitting.

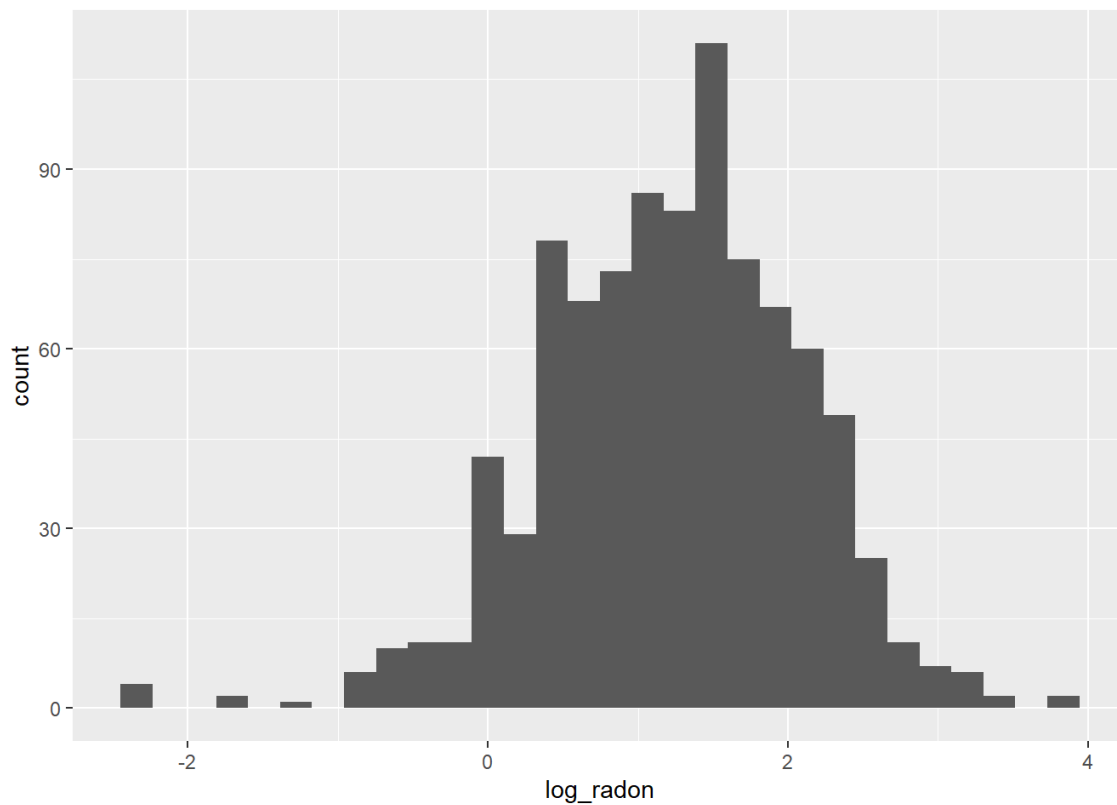
Multilevel modelling exercise

```
radon <- read.csv("radon.txt", header = T, sep = ",")
radon$county <- as.factor(radon$county)
```

Exercise 6: Do you think a hierachical model is warranted here? Do some EDA! Look for differences across counties.

```
# distribution of log_radon
ggplot(radon, aes(x=log_radon))+ geom_histogram()
```

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

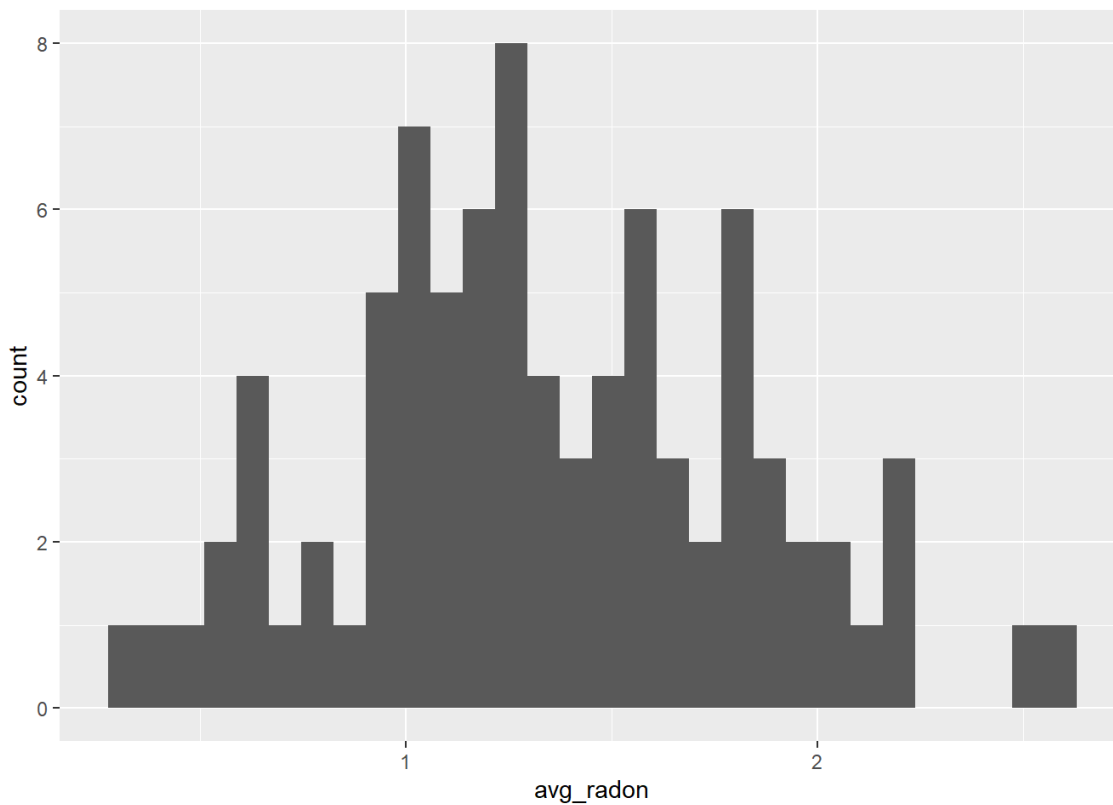


```
# for each county, calculate the sample average of the log_radon and plot the distribution of averages in a histogram
radon %>%
  group_by(county)%>%
  summarise(avg_radon = mean(log_radon)) %>%
  arrange(desc(avg_radon))%>%
  filter(row_number() ==1 | row_number() == n())
```

```
## # A tibble: 2 x 2
##   county      avg_radon
##   <fct>      <dbl>
## 1 LAC QUI PARLE  2.60
## 2 LAKE         0.322
```

```
radon %>%
  dplyr::group_by(county) %>%
  na.omit()%>%
  dplyr::summarise(avg_radon = mean(log_radon)) %>%
  dplyr::ungroup() %>%
  ggplot(aes(x = avg_radon))+
  geom_histogram()
```

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



I think hierachical model is warranted here, since distribution of log_radon is normal and distribution of average log_radon by county is also approximately normal.

Exercise 7: Begin by creating an unpooled model. Call that model `radon.unpooled`. Then create a hierachical/partially-pooled model where we model each county's intercept hierachically, again with no other predictors. Call that model `radon.mod1`.

```
radon.unpooled = stan_glm(data=radon,
                          formula = log_radon ~ -1 + county,
                          refresh = 0)
radon.mod1 = stan_glmer(data =radon,
                        formula = log_radon ~ 1 + (1|county),
                        refresh = 0)
```

```

n_county <- as.numeric(table(radon$county))
create_df <- function(sim,model){
  mean <- apply(sim,2,mean)
  sd <- apply(sim,2,sd)
  df <- cbind(n_county, mean, sd) %>%
    as.data.frame()%>%
    mutate(se = sd/ sqrt(n_county), model = model)
  return(df)
}

unpooled.sim <- as.matrix(radon.unpooled)
unpooled.df <- create_df(unpooled.sim[,1:85], model = "unpooled")

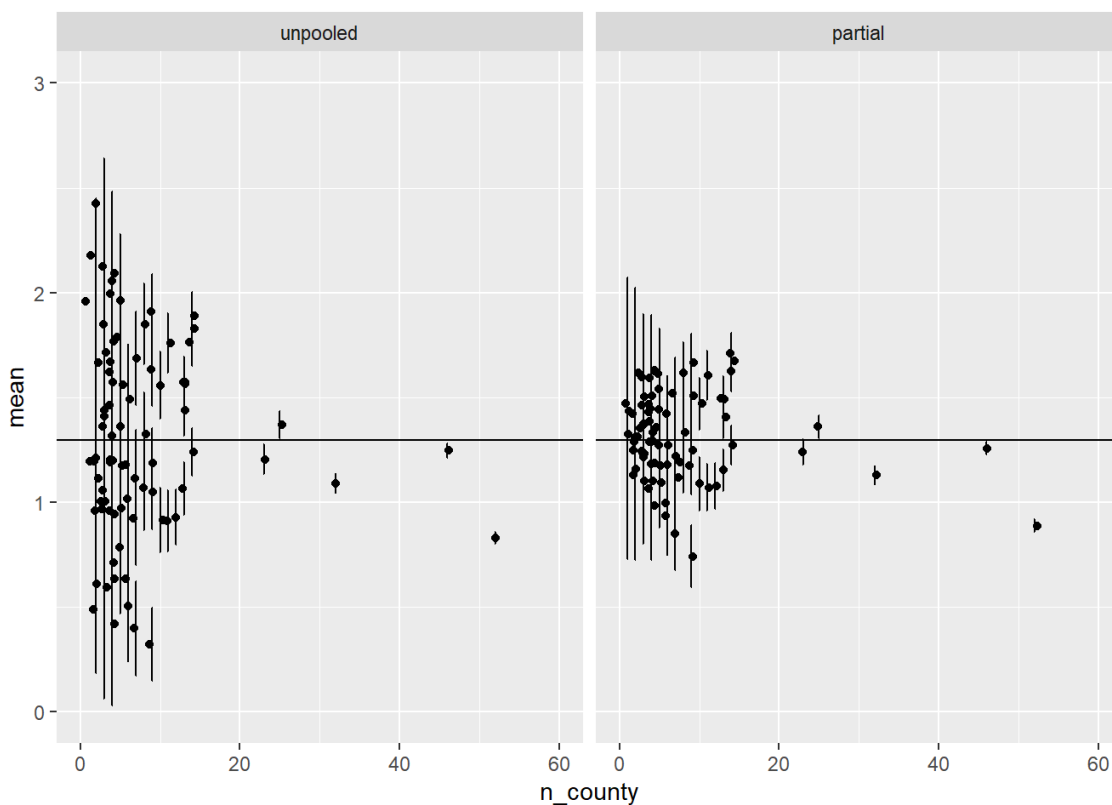
mod1.sim <- as.matrix(radon.mod1)[,1:86]
mod1.sim <- (mod1.sim[,1] + mod1.sim)[,-1]
partial.df <- create_df(mod1.sim, model = "partial")

ggplot(rbind(unpooled.df, partial.df)%>% mutate(model = factor(model, levels = c("unpooled", "partial"))), aes(x= n_county,
y = mean)) +
  #draws the means
  geom_jitter() +
  #draws the CI error bars
  geom_errorbar(aes(ymin=mean-2*se, ymax= mean+2*se), width=.1)+
  ylim(0,3)+
  xlim(0,60)+
  geom_hline(aes(yintercept= mean(coef(radon.unpooled))))+
  facet_wrap(~model)

```

```
## Warning: Removed 6 rows containing missing values (geom_point).
```

```
## Warning: Removed 12 rows containing missing values (geom_errorbar).
```



Exercise 8: Fit a varying-intercept model, but now add the variable floor as a fixed slope. Call the model `radon.mod2`. For another model `radon.mod3`, fit a varying-intercept and varying-slope model, again with floor as the predictor. Once you have fit these 5 models, report on the differences in model performance. Recall that we have a fourth variable which gives the county-level log uranium measurements. A really powerful aspect of hierarchical/multilevel modeling is the ability to incorporate data at different levels of coarseness. Fit a varying-intercept model, but include both floor and `log_uranium` in your model as well. So now we have both an individual/house-level covariate, as well as a group/county-level covariate. Group-level predictors help reduce group-level variation, which induces stronger pooling effects. Call this model `radon.mod4`.

```
radon.mod2 = stan_glmer(data=radon,
                        formula = log_radon ~ 1+ floor + (1|county),
                        refresh = 0)
radon.mod3 = stan_glmer(data = radon,
                        formula = log_radon ~ 1 + floor + (1+floor|county),
                        refresh = 0)
radon.mod4 = stan_glmer(data = radon,
                        formula = log_radon ~ 1 + floor + log_uranium +
                        (1 + floor + log_uranium | county),
                        refresh = 0)
```

```
## Warning: There were 1 divergent transitions after warmup. Increasing adapt_delta above 0.95 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
```

```
## Warning: Examine the pairs() plot to diagnose sampling problems
```

```
loo.unpooled = loo(radon.unpooled)
```

```
## Warning: Found 6 observation(s) with a pareto_k > 0.7. We recommend calling 'loo' again with argument 'k_threshold = 0.7'
in order to calculate the ELPD without the assumption that these observations are negligible. This will refit the model 6 ti
mes to compute the ELPDs for the problematic observations directly.
```

```
loo1 = loo(radon.mod1)
loo2 = loo(radon.mod2)
loo3 = loo(radon.mod3)
```

```
## Warning: Found 1 observation(s) with a pareto_k > 0.7. We recommend calling 'loo' again with argument 'k_threshold = 0.7'
in order to calculate the ELPD without the assumption that these observations are negligible. This will refit the model 1 ti
mes to compute the ELPDs for the problematic observations directly.
```

```
loo4 = loo(radon.mod4)
```

```
## Warning: Found 1 observation(s) with a pareto_k > 0.7. We recommend calling 'loo' again with argument 'k_threshold = 0.7'
in order to calculate the ELPD without the assumption that these observations are negligible. This will refit the model 1 ti
mes to compute the ELPDs for the problematic observations directly.
```

```
compare_models(loo.unpooled, loo1, loo2, loo3, loo4)
```

```
## Warning: 'compare_models' is deprecated.
## Use 'loo_compare' instead.
## See help("Deprecated")
```

```
## Warning: 'loo::compare' is deprecated.  
## Use 'loo_compare' instead.  
## See help("Deprecated")
```

```
## Model formulas:  
## : NULL  
## : NULL  
## : NULL  
## : NULL  
## : NULL  
##      elpd_diff se_diff  
## radon.mod4      0.0      0.0  
## radon.mod2     -7.3      5.6  
## radon.mod3     -8.9      5.3  
## radon.mod1    -54.8     12.3  
## radon.unpooled -79.8     14.3
```

radon.mod4 has the highest elpd_diff and radon.unpooled has the lowest.