

20/20

Applied Bayesian modeling - HW3

Álvaro J. Castro Rivadeneira

October 5, 2022

Score: The maximum number of points in this HW is 20 points, with 3 points extra credit. This HW counts for 1.5 HWs, so points are rescaled such that a “full pass” score is 150%. Specifically, for calculating your final HW score, the points will be rescaled to a maximum score of $(20+3)/20 \times 150\% = 172.5\%$.

Exercise 1: Fit a Bayesian model using brm and check and interpret the output (4 points) 4

We continue with IQ data, as introduced in HW2. For this HW, data set `iq_scores.csv` contains 10 iq-scores, sampled from a town in let's say, your favorite country.

```
iqs <- read_csv("iq_scores.csv", show_col_types = FALSE) %>%
  mutate(y = iq_score)
# Quick look at data
head(iqs)
```

```
## # A tibble: 6 x 2
##   iq_score     y
##   <dbl> <dbl>
## 1   114.  114.
## 2   130.  130.
## 3   121.  121.
## 4    94.4  94.4
## 5   112.  112.
## 6   116.  116.
```

```
summary(iqs)
```

```
##      iq_score      y
##  Min.   : 88.88  Min.   : 88.88
##  1st Qu.:102.52  1st Qu.:102.52
##  Median :112.96  Median :112.96
##  Mean   :111.06  Mean   :111.06
##  3rd Qu.:120.17  3rd Qu.:120.17
##  Max.   :129.86  Max.   :129.86
```

```
# Use a linear regression for estimates to compare with Bayesian estimates
iq.lm.fit <- lm(iq_score ~ 1, data = iqs)
summary(iq.lm.fit)
```

```
##
## Call:
## lm(formula = iq_score ~ 1, data = iqs)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22.176  -8.537   1.903   9.107  18.799
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  111.060      4.152   26.75  6.9e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.13 on 9 degrees of freedom

(iq.lm.mean.confint <- confint(iq.lm.fit, level = 0.8))

##              10 %      90 %
## (Intercept) 105.3179 116.8025

(iq.lm.sd <- sigma(iq.lm.fit))

## [1] 13.1297

# To calculate confidence intervals
n <- length(resid(iq.lm.fit))
k <- 2
alpha <- 0.2
(sigma <- summary(iq.lm.fit)$sigma)

## [1] 13.1297

(iq.lm.sd.low <- sqrt((n-(k+1))*sigma^2/qchisq(alpha/2, df = n-(k+1), lower.tail = FALSE)))

## [1] 10.02087

(iq.lm.sd.high <- sqrt((n-(k+1))*sigma^2/qchisq(1-alpha/2, df = n-(k+1), lower.tail = FALSE)))

## [1] 20.63823
```

Use brm to fit the following model to the IQ data:

$$\begin{aligned}
 y_i | \theta_i, \sigma^2 &\sim N(\theta_i, \sigma^2) (\text{independent}), \text{ for } i = 1, 2, \dots, n; \\
 \mu &\sim N(100, 15^2); \\
 \sigma &\sim \text{use the brm-default.}
 \end{aligned}$$

Bayesian regression

```
mu_prior <- set_prior("normal(100,15)", class = "Intercept")

iq.fit <- brm(iq_score ~ 1, data = iqs,
             prior = c(mu_prior),
             file = "output/exercise1",
             chains = 4, iter = 1000, warmup = 500,
             cores = getOption("mc.cores", 4))
```



Quick summary overview

```
summary(iq.fit)
```

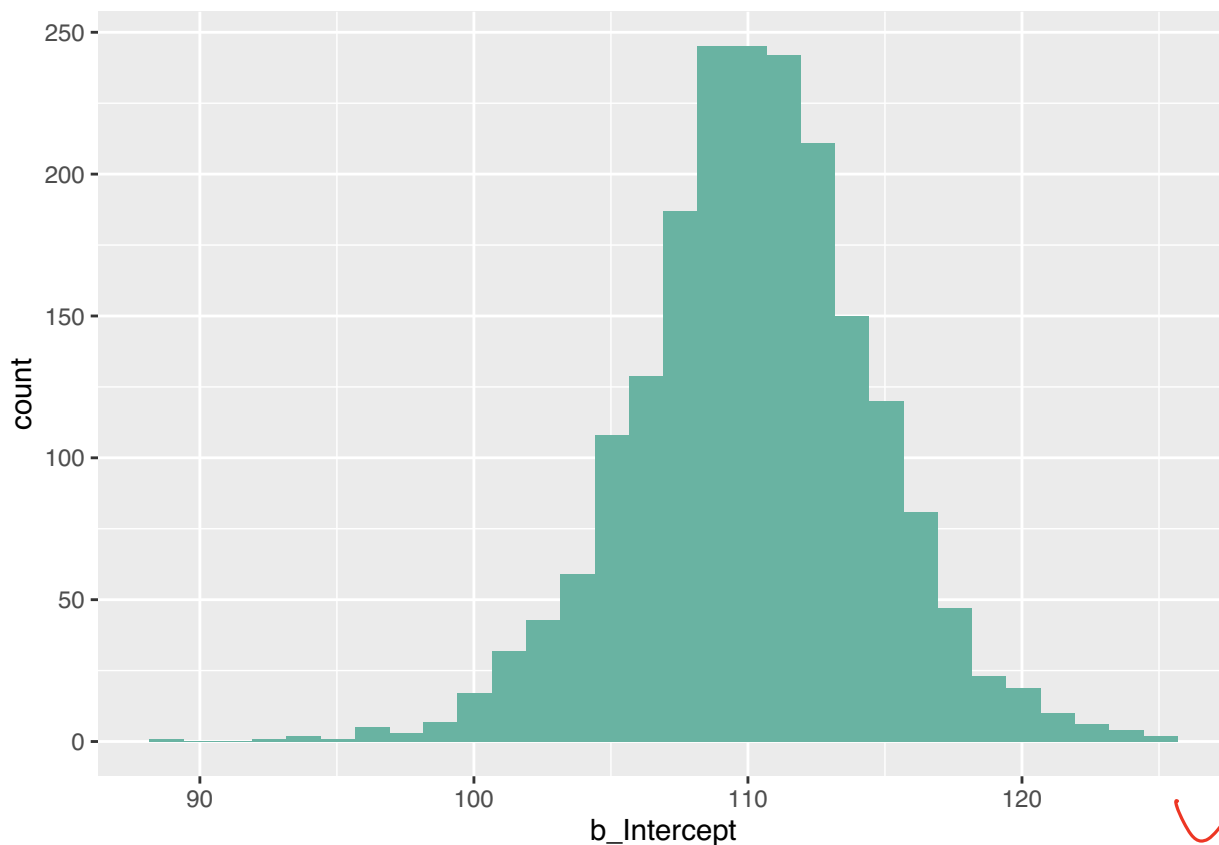
```
## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: iq_score ~ 1
## Data: iqs (Number of observations: 10)
## Draws: 4 chains, each with iter = 1000; warmup = 500; thin = 1;
## total post-warmup draws = 2000
##
## Population-Level Effects:
##      Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept  110.19      4.37  101.05  118.60 1.00    1109    891
##
## Family Specific Parameters:
##      Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    14.25      3.54   9.29   22.95 1.00    1283   1033
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

Then answer the following questions:

(i) Plot a histogram of the posterior samples of μ and report a posterior point estimate and 80% CI for μ .

```
# posterior_samples(iq.fit, pars = c("b_Intercept")) %>%
#   ggplot(aes(x = b_Intercept)) +
#   geom_histogram(bins=30) +
#   theme_bw()

post.iq.mean <- as_draws_df(iq.fit, variable = c("b_Intercept"))
post.iq.mean %>%
  ggplot(aes(x = b_Intercept)) +
  geom_histogram(bins=30, fill = "#69b3a2")
```



```
(post.iq.mean.summ <- posterior_summary(iq.fit, probs = c(.1, .9), variable = "b_Intercept"))
```

```
##           Estimate Est.Error      Q10      Q90
## b_Intercept 110.1903   4.368985 104.7669 115.5904
```

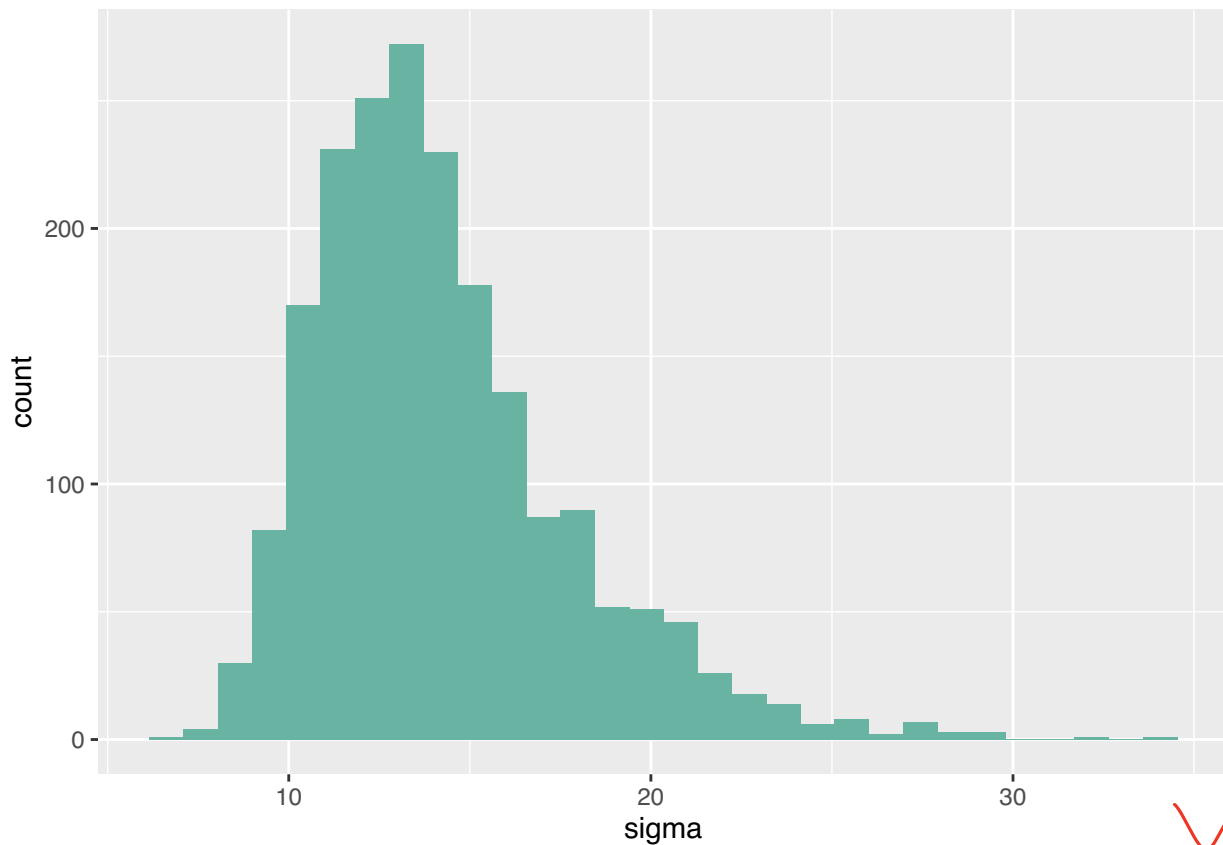
Answer

Posterior point estimate: 110.19, and the 80% CI for μ is 104.77 - 115.59. This is close to what was obtained with a simple linear model, and is slightly lower since the prior mean was also slightly lower than the data mean.

(ii) Plot a histogram of the posterior samples of σ and report a posterior point estimate and 80% CI for σ .

```
# posterior_samples(iq.fit, pars = c("sigma")) %>%
#   ggplot(aes(x = sigma)) +
#   geom_histogram(bins = 30) +
#   theme_bw()

post.iq.sigma <- as_draws_df(iq.fit, variable = c("sigma"))
post.iq.sigma %>%
  ggplot(aes(x = sigma)) +
  geom_histogram(bins=30, fill = "#69b3a2")
```



```
(post.iq.sig.summ <- posterior_summary(iq.fit, probs = c(.1, .9), variable = "sigma"))
```

```
##      Estimate Est.Error      Q10      Q90
## sigma 14.25081   3.537715 10.49514 19.11786
```

Answer

Posterior point estimate: 14.25, and the 80% CI for σ is 10.50 - 19.12. This is close to what was obtained with a simple linear model, where the estimate and 80% CI are: 13.13 (10.02 - 20.64).

(iii) Can you report a posterior point estimate and 80% CI for μ/σ ? If yes, do so. If not, why not?

Answer

Yes. I can calculate the ratio from my posterior samples as follows:

```
post.iq.meandivsigma <- post.iq.mean$b_Intercept/post.iq.sigma$sigma
(post.iq.meandivsig.summ <- posterior_summary(post.iq.meandivsigma, probs = c(.1, .9)))
```

```
##      Estimate Est.Error      Q10      Q90
## [1,] 8.167348   1.864061  5.726022 10.53436
```

```
# Check that it is approximately correct by getting ratio from previously calculated point estimates
post.iq.mean.summ[1]/post.iq.sig.summ[1]
```

```
## [1] 7.732213
```

```
# It's close, which gives me confidence that it was done correctly
```

The reason it is possible is that when I produce my model fit, I'm producing samples that include μ and σ , so that I can calculate the ratio of each of these samples and then calculate a point estimate and credible intervals based on their distribution. ✓

Note that the prior for μ can be specified with an additional argument in `brm` (as illustrated in optional material in module 5), as follows:

```
#mu_prior <- set_prior("normal(100,15)", class = "Intercept")  
#fit <- brm(y ~ 1, prior = c(mu_prior),  
#           your other usual arguments)
```

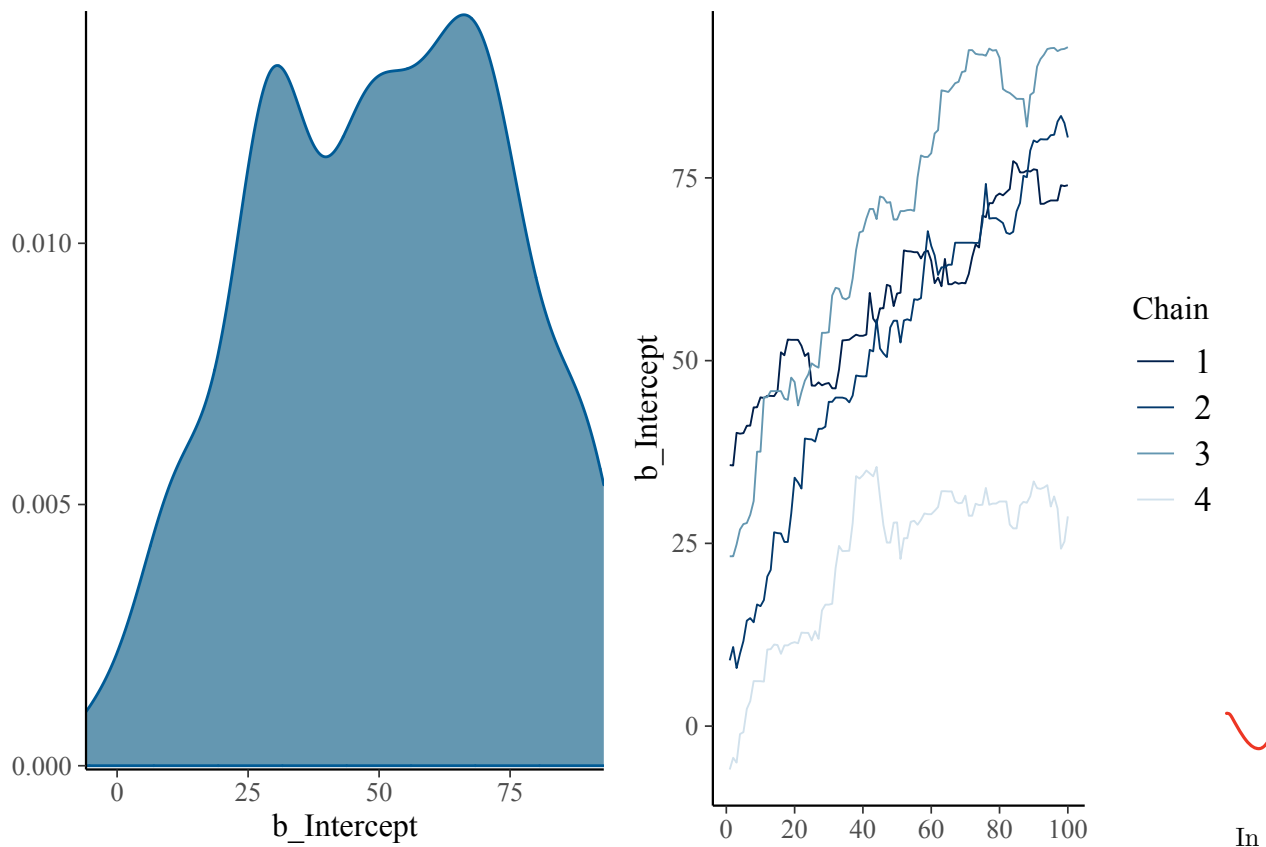
Exercise 2: Compare and contrast the MCMC diagnostics of two different model fits (4 points) (4)

Continue with the IQ data from Q1 (with y = IQ scores) to fit the model as specified below. Present and briefly summarize resulting MCMC diagnostics for μ (traceplots, Rhat, effective sample sizes). Then and comment on whether this model fit can be used for summarizing information regarding μ . If not, why not?

```
iq.fit_bad <- brm(iq_score ~ 1, data = iqs, file = "output/exercise2",  
                 chains = 4, iter = 200, cores = getOption("mc.cores", 4),  
                 control = list(adapt_delta = 0.6, max_treedepth = 4)  
                 # these are NOT recommended options, trying to create problems here!  
)
```

MCMC diagnostics for μ ✓

```
# plot(iq.fit, variable = c("b_Intercept"))  
plot(iq.fit_bad, variable = c("b_Intercept"))
```



this plot we see that the estimate for μ is first of all, wrong - it is much lower than our prior estimate. Moreover, looking at the distribution on the left, the data do not seem to be normally distributed. More importantly, looking at the plot on the right, we don't see the chains converging, and in fact chain 4 seems to diverge from the others. Further, there is no mixing, and the autocorrelation in the sampled values is high.

right?

```
# summary(iq.fit)
summary(iq.fit_bad)
```

```
## Warning: Parts of the model have not converged (some Rhats are > 1.05). Be
## careful when analysing the results! We recommend running more iterations and/or
## setting stronger priors.
```

```
## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: iq_score ~ 1
## Data: iqs (Number of observations: 10)
## Draws: 4 chains, each with iter = 200; warmup = 100; thin = 1;
## total post-warmup draws = 400
##
## Population-Level Effects:
##      Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    50.65    23.83    6.16   92.42 2.36         5      19
##
## Family Specific Parameters:
##      Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma    58.41    24.16    20.63   113.58 1.73         7      15
##
```

```
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

From these results, it is first obvious that they differ significantly from what we obtained in the previous exercise as well as what we found with a linear model. Further, we notice that Rhat is 2.36, which is much, much higher than the target value where $\hat{R} < 1.05$. Additionally, the ESS should be at least 400 (100 per Markov Chain), and in fact it is 5 for the Bulk-ESS and 19 for the Tail-ESS.

Ultimately this means that this model fit can *not* be used for summarizing information regarding μ because it does not adequately converge in a way that provides meaningful results. The chains did not mix well, and have a very low precision as judged by S_{eff} .

Exercise 3: (8 points)

8

Background: The dataset “marriage.csv” contains (simulated) data on the age of first marriage for a number of women in Kenya. Information on the ethnic group is provided as well. For the questions below, let y_i denote the age of first marriage for the i -th woman, and $j[i]$ her ethnic group. The goal is to learn more about the mean age of marriage within ethnic groups, using a Bayesian hierarchical model.

For starters, I will look at the data:

```
matri <- read_csv("marriage.csv", show_col_types = FALSE) %>%
  mutate(y = agemarried,
         j = ethnicgroup)
# Quick look at data
head(matri)
```

```
## # A tibble: 6 x 4
##   agemarried ethnicgroup     y     j
##   <dbl>      <dbl> <dbl> <dbl>
## 1      20.9          1  20.9     1
## 2      24.7          1  24.7     1
## 3      19.4          2  19.4     2
## 4      19.5          2  19.5     2
## 5      21.1          3  21.1     3
## 6      17.4          3  17.4     3
```

```
summary(matri)
```

```
##   agemarried   ethnicgroup     y     j
##   Min.   :10.73   Min.    : 1.00   Min.   :10.73   Min.    : 1.00
##   1st Qu.:18.36   1st Qu.:24.00   1st Qu.:18.36   1st Qu.:24.00
##   Median :20.01   Median :30.00   Median :20.01   Median :30.00
##   Mean   :19.94   Mean    :29.39   Mean    :19.94   Mean    :29.39
##   3rd Qu.:21.51   3rd Qu.:35.00   3rd Qu.:21.51   3rd Qu.:35.00
##   Max.   :28.40   Max.    :40.00   Max.    :28.40   Max.    :40.00
```

```
# Create summary dataset with info for each ethnic group:
matrigrup <- matri %>%
  group_by(ethnicgroup) %>%
  summarise(npers = n(), ybar = mean(y))
matrigrup
```

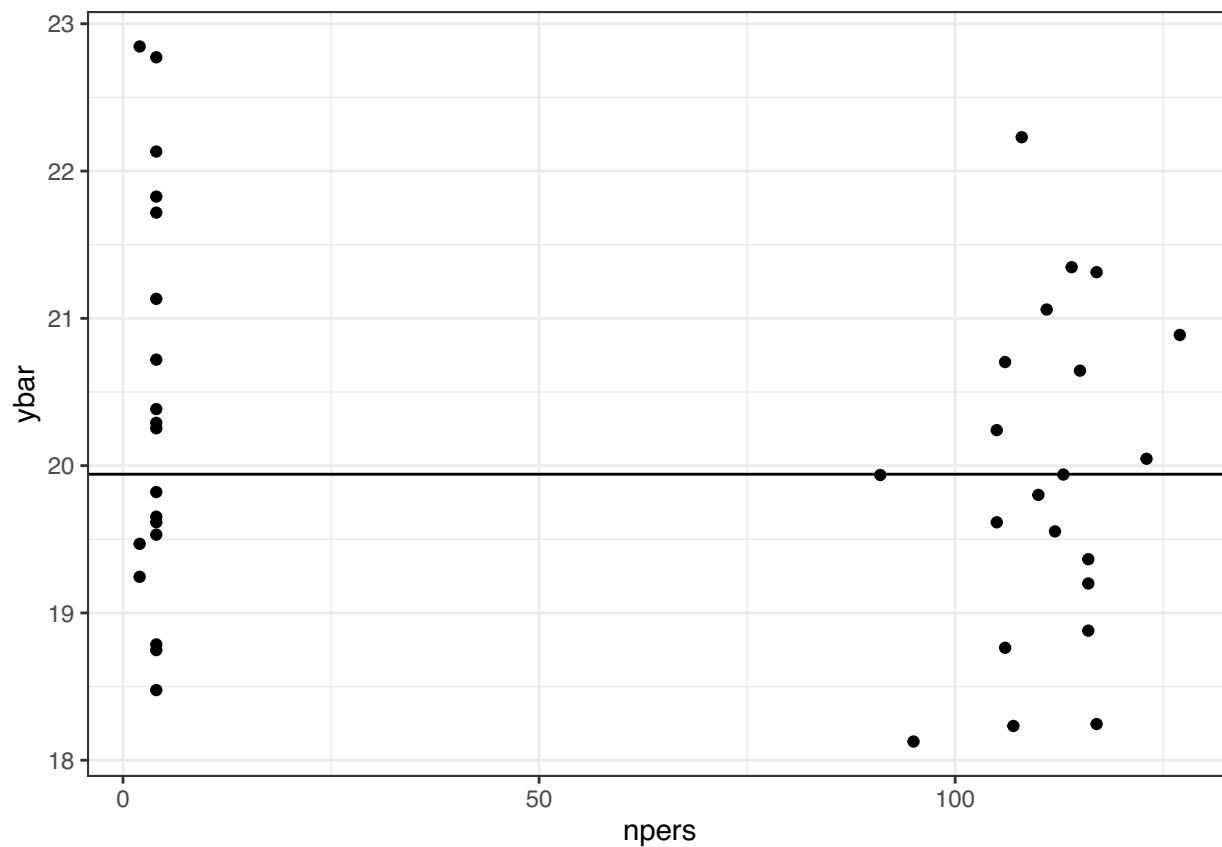


```
## # A tibble: 40 x 3
##   ethnicgroup npers  ybar
##   <dbl> <int> <dbl>
## 1         1     2  22.8
## 2         2     2  19.5
## 3         3     2  19.2
## 4         4     4  19.8
## 5         5     4  19.5
## 6         6     4  18.7
## 7         7     4  20.7
## 8         8     4  21.8
## 9         9     4  20.3
## 10        10     4  22.8
## # ... with 30 more rows
```

```
summary(matrigrup)
```

```
##   ethnicgroup      npers      ybar
##  Min.   : 1.00   Min.   : 2.0   Min.   :18.13
## 1st Qu.:10.75   1st Qu.: 4.0   1st Qu.:19.33
##  Median :20.50   Median : 93.0   Median :19.94
##  Mean   :20.50   Mean   : 60.0   Mean   :20.14
## 3rd Qu.:30.25   3rd Qu.:112.2   3rd Qu.:20.93
##  Max.   :40.00   Max.   :127.0   Max.   :22.85
```

```
# population mean
ybarbar <- mean(matri$y)
matrigrup %>%
  ggplot(aes(x = npers, y = ybar)) +
  geom_point() +
  geom_hline(mapping = aes(yintercept = ybarbar)) +
  theme_bw()
```



I decided to use multilevel linear modeling for estimates to compare with Bayesian estimates

```
# First I will make a simple linear model without coefficients to compare with the hierarchical model
matrisimple.lm.fit <- lm(agemarried ~ 1, data=matri)
summary(matrisimple.lm.fit)
```

```
##
## Call:
## lm(formula = agemarried ~ 1, data = matri)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -9.2136 -1.5777  0.0716  1.5678  8.4615
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  19.9416     0.0465   428.9  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.278 on 2399 degrees of freedom
```

```
confint(matrisimple.lm.fit)
```

```
##              2.5 %    97.5 %
## (Intercept) 19.85046 20.03281
```

```
# Now the hierarchical model
```

```
matri.lm.fit <- lmer(agemarried ~ (1 | ethnicgroup), data=matri)  
summary(matri.lm.fit)
```

```
## Linear mixed model fit by REML ['lmerMod']  
## Formula: agemarried ~ (1 | ethnicgroup)  
## Data: matri  
##  
## REML criterion at convergence: 10251.3  
##  
## Scaled residuals:  
##      Min       1Q   Median       3Q      Max   
## -3.7610 -0.6852 -0.0062  0.6993  3.7065   
##  
## Random effects:  
## Groups      Name      Variance Std.Dev.  
## ethnicgroup (Intercept) 1.112    1.054  
## Residual                4.046    2.011  
## Number of obs: 2400, groups: ethnicgroup, 40  
##  
## Fixed effects:  
##              Estimate Std. Error t value  
## (Intercept)  20.0613    0.1933   103.8
```

```
(matri.lm.mean <- fixef(matri.lm.fit))
```

```
## (Intercept)  
##      20.06132
```

```
(matri.lm.mean.confint <- confint(matri.lm.fit, level = 0.95)[3,])
```

```
## Computing profile confidence intervals ...
```

```
##      2.5 %   97.5 %  
## 19.68137 20.44960
```

```
# eta = alpha - mu_alpha
```

```
eta.lm.1 <- as_tibble(ranef(matri.lm.fit)$ethnicgroup, rownames = "ethnicgroup")  
head(eta.lm.1)
```

```
## # A tibble: 6 x 2  
## ethnicgroup `(Intercept)`  
##   <chr>          <dbl>  
## 1 1             0.988  
## 2 2            -0.210  
## 3 3            -0.290  
## 4 4            -0.126  
## 5 5            -0.278  
## 6 6            -0.688
```

```
eta.lm.2 <- as.data.frame(ranef(matri.lm.fit)) %>%
  transform(lwr = condval - 1.96*condsd, upr = condval + 1.96*condsd)
head(eta.lm.2)
```

```
##      grpvar      term grp  condval  condsd      lwr      upr
## 1 ethnicgroup (Intercept) 1  0.9876779 0.8470632 -0.6725659 2.6479217
## 2 ethnicgroup (Intercept) 2 -0.2101617 0.8470632 -1.8704055 1.4500821
## 3 ethnicgroup (Intercept) 3 -0.2895341 0.8470632 -1.9497779 1.3707097
## 4 ethnicgroup (Intercept) 4 -0.1260259 0.7277684 -1.5524520 1.3004002
## 5 ethnicgroup (Intercept) 5 -0.2775169 0.7277684 -1.7039430 1.1489092
## 6 ethnicgroup (Intercept) 6 -0.6878874 0.7277684 -2.1143135 0.7385387
```

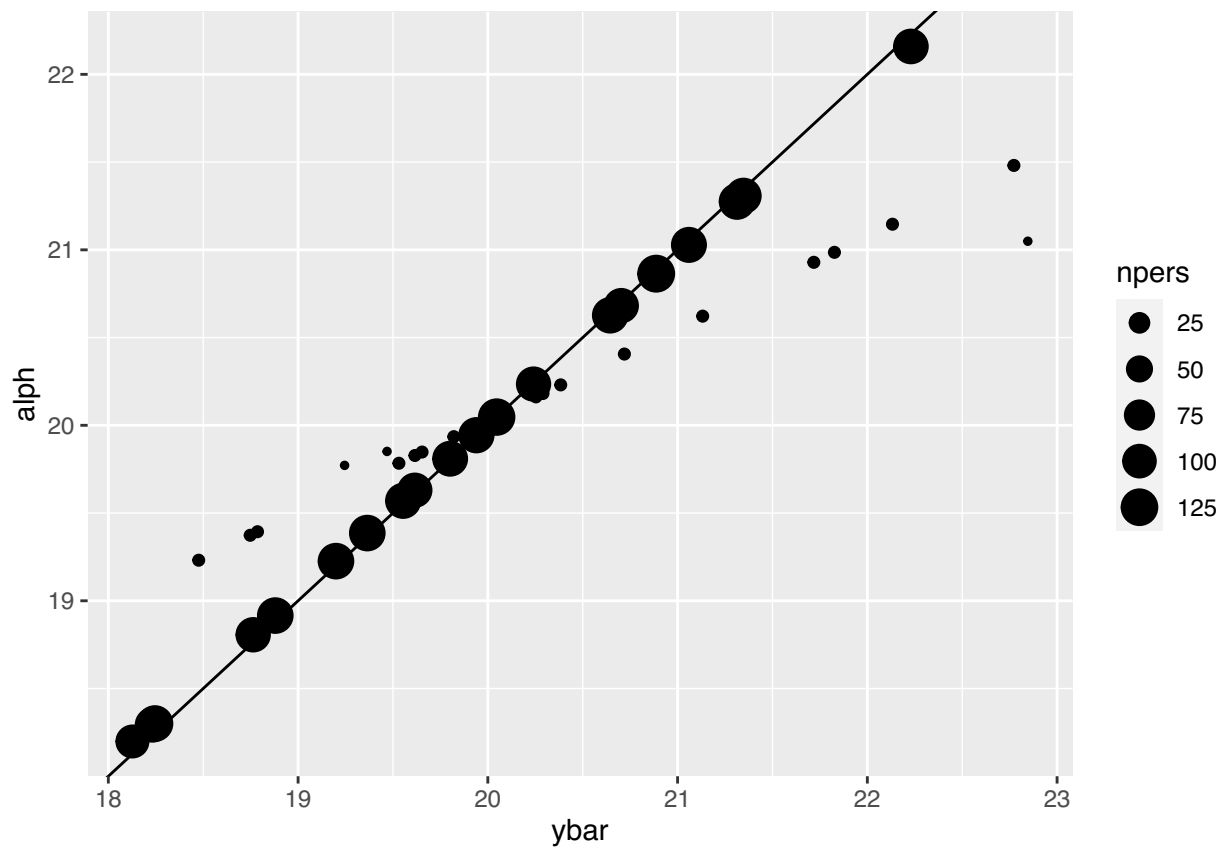
```
eta.lm <- broom.mixed::tidy(matri.lm.fit, effects = "ran_vals", conf.int = TRUE)
# alphas
lm.alphas <- coef(matri.lm.fit, summary = T)[1]$ethnicgroup %>%
  as_tibble(rownames = "ethnicgroup") %>%
  rename(alph = "(Intercept)") %>%
  transform(ethnicgroup = as.integer(ethnicgroup))
head(lm.alphas)
```

```
## ethnicgroup  alph
## 1          1 21.04900
## 2          2 19.85116
## 3          3 19.77179
## 4          4 19.93530
## 5          5 19.78381
## 6          6 19.37344
```

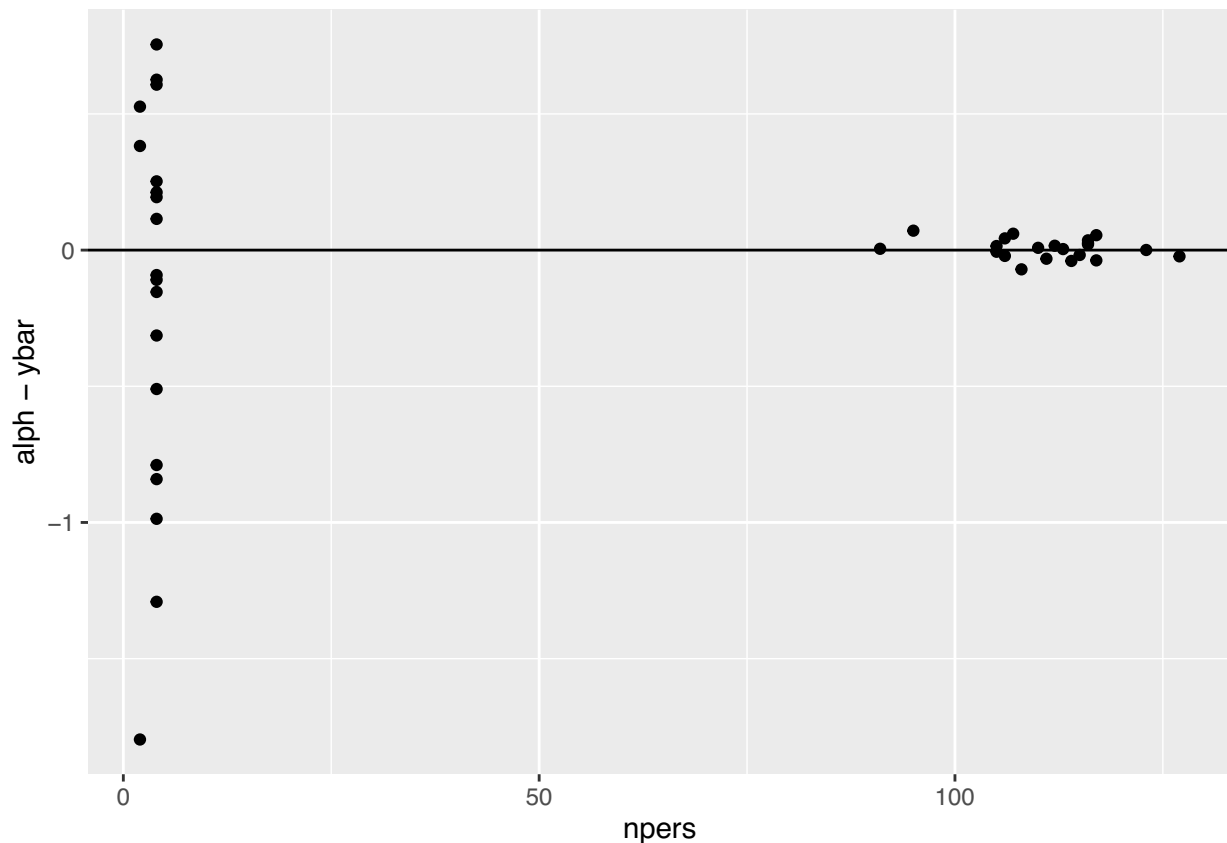
```
lm.alphas.full <- eta.lm %>%
  rename(ethnicgroup = level) %>%
  transform(ethnicgroup = as.integer(ethnicgroup)) %>%
  left_join(lm.alphas, by = "ethnicgroup") %>%
  select(3,5:9) %>%
  mutate(estimate = (matri.lm.mean + estimate),
         conf.low = (conf.low + matri.lm.mean),
         conf.high = (conf.high + matri.lm.mean))
head(lm.alphas.full)
```

```
## ethnicgroup estimate std.error conf.low conf.high  alph
## 1          1 21.04900 0.8470632 19.38879 22.70922 21.04900
## 2          2 19.85116 0.8470632 18.19095 21.51138 19.85116
## 3          3 19.77179 0.8470632 18.11158 21.43200 19.77179
## 4          4 19.93530 0.7277684 18.50890 21.36170 19.93530
## 5          5 19.78381 0.7277684 18.35741 21.21021 19.78381
## 6          6 19.37344 0.7277684 17.94704 20.79984 19.37344
```

```
# Make the plot of alpha ~ ybar
matrigruplmalpha <- lm.alphas %>%
  left_join(matrigrup, by = "ethnicgroup")
matrigruplmalpha %>%
  ggplot(aes(y = alph, x = ybar, size = npers)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0)
```



```
# Plot of alpha - ybar
matrigruplmalpha %>%
  ggplot(aes(y = alph - ybar, x = npers)) +
  geom_point() +
  geom_hline(yintercept = 0)
```



To answer the questions below, fit the following Bayesian hierarchical model to the marriage data:

$$y_i | \alpha_{j[i]}, \sigma_y \stackrel{i.i.d.}{\sim} N(\alpha_{j[i]}, \sigma_y^2),$$

$$\alpha_j | \mu_\alpha, \sigma_\alpha \stackrel{i.i.d.}{\sim} N(\mu_\alpha, \sigma_\alpha^2),$$

using default priors for $\mu_\alpha, \sigma_\alpha, \sigma_y$ as set in the brm function. In the questions below, if you present default brm-output, please indicate what brm-parameter name refers to what greek letter in the equations above.

```
# Fixed some issues with the model (increased warmup and iterations), after examining diagnostics
matri.fit <- brm(y ~ (1|ethnicgroup), data = matri,
  file = "output/exercise3",
  chains = 4, iter = 2000, warmup = 1000,
  cores = getOption("mc.cores", 4))
```

- (a) State the point estimates and 95% CIs of each of the following parameters and interpret (explain what information is given by) these estimates: $\mu_\alpha, \sigma_\alpha, \sigma_y$.

```
# Reviewing output and diagnostics
summary(matri.fit)
```

```
## Family: gaussian
## Links: mu = identity; sigma = identity
## Formula: y ~ (1 | ethnicgroup)
## Data: matri (Number of observations: 2400)
## Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
## total post-warmup draws = 4000
```

```
##
## Group-Level Effects:
## ~ethnicgroup (Number of levels: 40)
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)    1.08      0.15    0.83    1.41 1.00      489    1401
##
## Population-Level Effects:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept    20.05      0.20   19.68   20.46 1.00      501      726
##
## Family Specific Parameters:
##           Estimate Est.Error 1-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sigma        2.01      0.03    1.96    2.07 1.00     4502     2687
##
## Draws were sampled using sampling(NUTS). For each parameter, Bulk_ESS
## and Tail_ESS are effective sample size measures, and Rhat is the potential
## scale reduction factor on split chains (at convergence, Rhat = 1).
```

```
# names(summary(matri.fit))
summary(matri.fit)$fixed
```

```
##           Estimate Est.Error 1-95% CI u-95% CI      Rhat Bulk_ESS Tail_ESS
## Intercept 20.05364  0.200092 19.67731 20.46442 1.002948 500.8331 725.6554
```

```
# mu_alpha
fixef(matri.fit)
```

```
##           Estimate Est.Error      Q2.5      Q97.5
## Intercept 20.05364  0.200092 19.67731 20.46442
```

```
mu_alpha <- fixef(matri.fit)[1]
# sigma_alpha
summary(matri.fit)$random
```

```
## $ethnicgroup
##           Estimate Est.Error 1-95% CI u-95% CI      Rhat Bulk_ESS Tail_ESS
## sd(Intercept) 1.081793 0.1529015 0.8252853 1.411906 1.004535 489.1233 1400.779
```

```
# sigma_y
summary(matri.fit)$spec_pars
```

```
##           Estimate Est.Error 1-95% CI u-95% CI      Rhat Bulk_ESS Tail_ESS
## sigma 2.012571 0.03015565 1.955434 2.072554 1.002573 4502.107 2687.056
```

After correcting some of the settings (increasing the warmup and iterations), our diagnostics are reasonable, with Rhats of 1.00 and ESS results between 489 and 4,502. With reference to the equations above, “Intercept” refers to α_j , with a mean of the group means (the grand intercept to quote Bürkner) μ_α of 20.05 and a 95% CI of the posterior distribution of 19.68 - 20.46, which can be interpreted as saying that there is a 0.95 probability that the value of μ_α is in that interval. Our results also show a standard deviation σ_α of 1.08, which is the standard deviation of the distribution of varying intercepts across ethnic groups. This has a CI of 0.83 - 1.41, with an interpretation similar to that of μ_α . The residual standard deviation, σ_y is the error of each y_i around each group mean μ_i , and is 2.01, with a CI of 1.96 - 2.07. Each ethnic group has its own group mean (intercept) $\alpha_{j[i]}$ and these are:

```

# eta = alpha - mu_alpha
eta <- as_tibble(ranef(matri.fit)$ethnicgroup[, "Intercept"], rownames = "ethnicgroup")
# alpha = eta + mu_alpha
alphas <- coef(matri.fit, summary = T)$ethnicgroup %>%
  as_tibble(rownames = "ethnicgroup") %>%
  rename(alpha = Estimate.Intercept) %>%
  transform(ethnicgroup = as.integer(ethnicgroup))
matrigrupalpha <- alphas %>%
  left_join(matrigrup, by = "ethnicgroup")
matrigrupalpha

```

##	ethnicgroup	alph	Est.Error.Intercept	Q2.5.Intercept	Q97.5.Intercept
## 1	1	21.06499	0.9028678	19.32025	22.86666
## 2	2	19.85901	0.8789042	18.11652	21.57998
## 3	3	19.75798	0.8697712	18.00896	21.48954
## 4	4	19.92870	0.7298724	18.48644	21.35747
## 5	5	19.76674	0.7510800	18.27095	21.25708
## 6	6	19.37364	0.7503681	17.89985	20.81579
## 7	7	20.40112	0.7349098	18.99257	21.89171
## 8	8	20.96369	0.7503440	19.53352	22.46732
## 9	9	20.14094	0.7579572	18.65003	21.60098
## 10	10	21.49447	0.7699781	20.01317	23.05629
## 11	11	19.36764	0.7679026	17.88570	20.87540
## 12	12	21.14230	0.7674816	19.68152	22.69853
## 13	13	20.17800	0.7276783	18.77459	21.63106
## 14	14	19.19575	0.7344050	17.73694	20.59866
## 15	15	20.61698	0.7299643	19.19219	21.99853
## 16	16	19.81858	0.7374639	18.34767	21.23685
## 17	17	19.82764	0.7516008	18.31427	21.32216
## 18	18	20.22987	0.7534170	18.75189	21.74411
## 19	19	20.93295	0.7442662	19.53599	22.41512
## 20	20	19.22821	0.1853276	18.86759	19.59008
## 21	21	19.56918	0.1893589	19.19877	19.94546
## 22	22	19.94732	0.1877737	19.58658	20.31793
## 23	23	21.02765	0.1809678	20.67325	21.38043
## 24	24	22.16002	0.1944254	21.77895	22.53807
## 25	25	21.30837	0.1841621	20.94628	21.66503
## 26	26	20.68068	0.1907830	20.30998	21.05129
## 27	27	18.19482	0.2049113	17.79856	18.59652
## 28	28	20.86303	0.1787073	20.51211	21.22145
## 29	29	20.62584	0.1858777	20.25733	20.99452
## 30	30	19.62795	0.1947698	19.25527	20.01279
## 31	31	19.38683	0.1836266	19.03540	19.74475
## 32	32	18.91619	0.1844097	18.55296	19.27900
## 33	33	19.80767	0.1894349	19.43639	20.17381
## 34	34	20.23586	0.1967650	19.85245	20.61717
## 35	35	18.29880	0.1843892	17.94769	18.65998
## 36	36	18.29163	0.1958576	17.90965	18.68324
## 37	37	20.04606	0.1749500	19.70291	20.38628
## 38	38	18.80590	0.1977433	18.41094	19.20045
## 39	39	19.94033	0.2097117	19.52562	20.36338
## 40	40	21.27292	0.1832710	20.91947	21.63045
##	npers	ybar			


```
## 1      2 22.84582
## 2      2 19.46883
## 3      2 19.24506
## 4      4 19.82066
## 5      4 19.53137
## 6      4 18.74772
## 7      4 20.71963
## 8      4 21.82662
## 9      4 20.25360
## 10     4 22.77228
## 11     4 18.78626
## 12     4 22.13255
## 13     4 20.29049
## 14     4 18.47623
## 15     4 21.13214
## 16     4 19.61580
## 17     4 19.65328
## 18     4 20.38408
## 19     4 21.71790
## 20    116 19.19971
## 21    112 19.55367
## 22    113 19.93968
## 23    111 21.05987
## 24    108 22.22955
## 25    114 21.34729
## 26    106 20.70314
## 27     95 18.12697
## 28    127 20.88714
## 29    115 20.64484
## 30    105 19.61524
## 31    116 19.36472
## 32    116 18.87984
## 33    110 19.80131
## 34    105 20.24095
## 35    117 18.24566
## 36    107 18.23270
## 37    123 20.04645
## 38    106 18.76331
## 39     91 19.93633
## 40    117 21.31348
```

(b) State the point estimate and 95% CIs for α_1 and interpret (explain what information is given by) the estimate.

This was estimated previously and is given by:

```
alphas[1,]
```

```
## ethnicgroup      alph Est.Error.Intercept Q2.5.Intercept Q97.5.Intercept
## 1              1 21.06499             0.9028678         19.32025         22.86666
```

```
# Interesting to compare with linear model estimate
lm.alphas.full[1,]
```

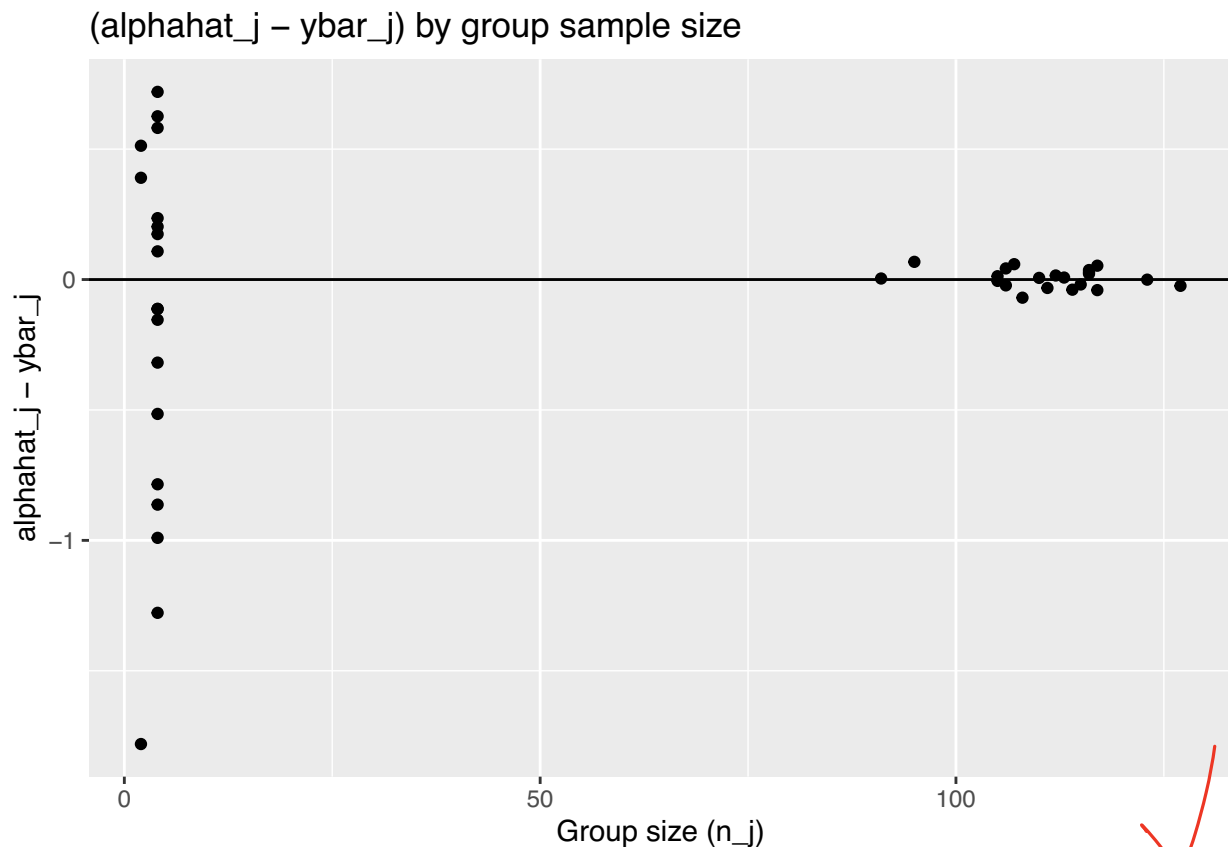
```
## ethnicgroup estimate std.error conf.low conf.high alph
## 1 1 21.049 0.8470632 19.38879 22.70922 21.049
```

They are very similar, although the CI is slightly more narrow with the linear model.

Thus, ethnic group 1 has a point estimate mean of 21.06, with a 95% CI of 19.32 - 22.87, which means the partially pooled (shrunk) estimate for the mean marriage age of ethnic group one is 21.06, with 95% CI given before. ✓

- (c) Construct two plots: (a) plot of $\hat{\alpha}_j - \bar{y}_j$ against the within-ethnic-group sample size n_j and (b) plot $\hat{\alpha}_j$ against \bar{y}_j , with the identity line added. Explain what information these plots provide regarding the comparison of \bar{y}_j and $\hat{\alpha}_j$ for estimating the mean age of marriage within ethnic groups.

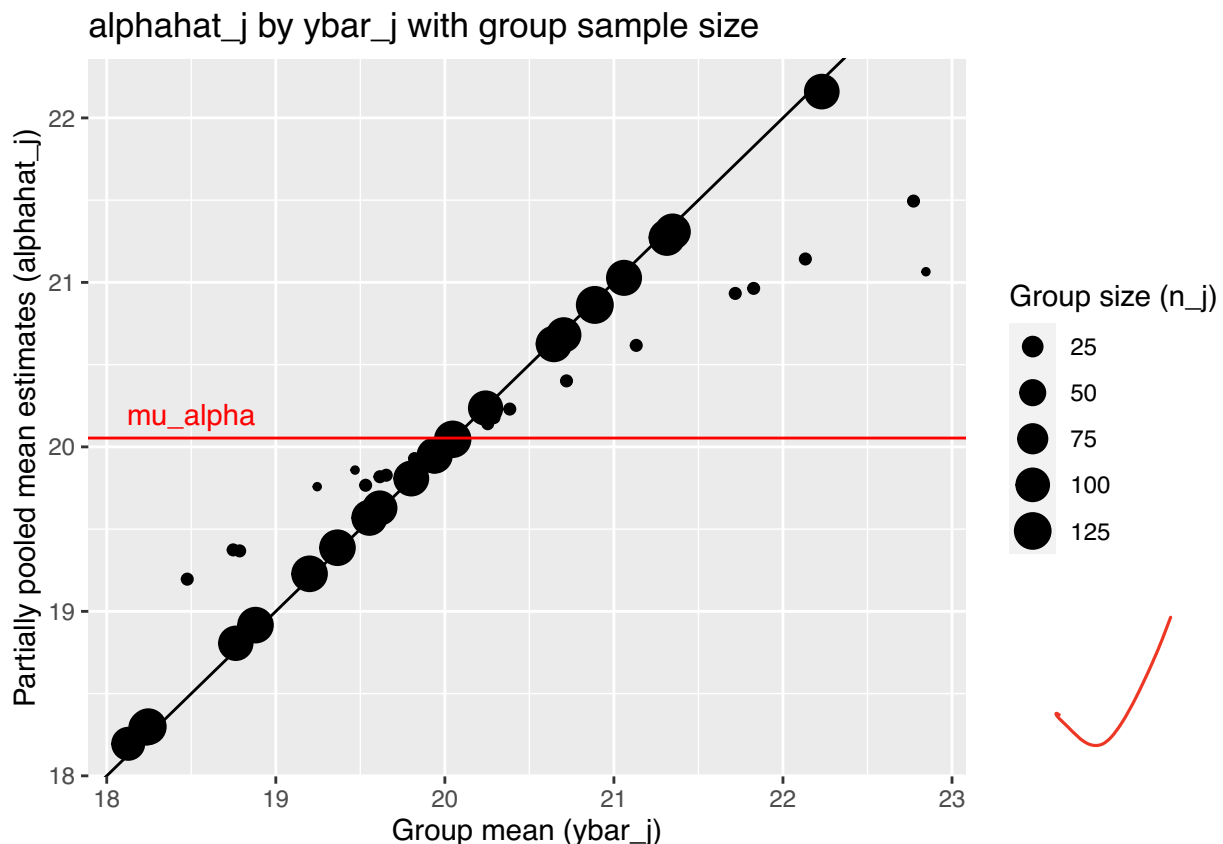
```
# Plot of alpha - ybar
matrigrupalpha %>%
  ggplot(aes(y = alph - ybar, x = npers)) +
  geom_point() +
  geom_hline(yintercept = 0) +
  labs(title = "(alphahat_j - ybar_j) by group sample size",
       x = "Group size (n_j)",
       y = "alphahat_j - ybar_j")
```



```

# Plot of alpha ~ ybar
matrigrupalpha %>%
  ggplot(aes(y = alph, x = ybar, size = npers)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0) +
  geom_abline(slope = 0, intercept = mu_alpha, color = "red", ) +
  labs(title = "alphahat_j by ybar_j with group sample size",
       x = "Group mean (ybar_j)",
       y = "Partially pooled mean estimates (alphahat_j)",
       size = "Group size (n_j)") +
  theme(legend.position = "right") +
  annotate("text", x = 18.5, y = 20.2,
         label = "mu_alpha", color = "red"
  )

```



The first plot is showing that for large groups (large n_j), the difference between the partially pooled estimates for the mean and the actual mean of each group is very small, as the weight given by the large sample size means that \bar{y}_j and $\hat{\alpha}_j$ are very close. Whereas in small groups, there is great variability, and the difference between \bar{y}_j and $\hat{\alpha}_j$ can be much larger as the overall group mean μ_α has more weight in calculating the partially pooled mean.

The second plot, similarly shows that in large groups \bar{y}_j and $\hat{\alpha}_j$ are almost equal (seen in the proximity to the identity line), since the large sample size means that more weight is given to \bar{y}_j in calculating the partially pooled mean. Whereas, for smaller groups, μ_α has greater weight and pulls the data in its direction, which can be seen by how they are further from the identity, but closer to μ_α .

Optional/just for fun exercise: As stated in the slides, for the multilevel model under consideration here, the full conditional distribution for the j -th state mean is given by:

$$\alpha_j | \mathbf{y}, \mu_\alpha, \sigma_y, \sigma_\alpha \sim N(m, v),$$

$$v = (n_j / \sigma_y^2 + 1 / \sigma_\alpha^2)^{-1},$$

$$m = v \cdot \left(\frac{n_j}{\sigma_y^2} \bar{y} + \frac{1}{\sigma_\alpha^2} \mu_\alpha \right) = \frac{\frac{n_j}{\sigma_y^2} \bar{y} + \frac{1}{\sigma_\alpha^2} \mu_\alpha}{\frac{n_j}{\sigma_y^2} + \frac{1}{\sigma_\alpha^2}}$$

Give the derivation of the full conditional.

Hint: Check slide in part 1 for the derivation of the normal distribution for μ in the normal-normal setting.

Exercise 4 (4 points) 4

Continue with the marriage data set from exercise 3.

The goal in this exercise is to predict the age at first marriage for a randomly sampled woman in an ethnic group for which we have not yet observed any data, using the model and data from exercise 3.

- (a) Obtain samples from the predictive posterior density for the age at first marriage and visualize the samples in a histogram. In your answer, include R code (that does NOT use the predict function from brms) as well as a write up in equations how you obtained the samples. Make sure to introduce notation first to explain what you're sampling.

First the notation: We assume that for our prediction, the hierarchical sampling distribution holds true, where we want to predict the age at first marriage for a randomly sampled woman k , in a new, unsampled ethnic group $h = j[k]$, that follows the distribution:

$$\tilde{y}_k | \tilde{\alpha}_{j[k]}, \sigma_y^2 \sim N(\tilde{\alpha}_{j[k]}, \sigma_y^2),$$

$$\tilde{\alpha}_h | \mu_\alpha, \sigma_\alpha^2 \sim N(\mu_\alpha, \sigma_\alpha^2),$$

with other parameters previously defined. So, I will obtain random samples for $\tilde{\alpha}_h$ using the parameters for μ_α and σ_α^2 obtained from the posterior Bayesian fit of our model. In other words, I will use the 4,000 samples of μ_α and σ_α^2 for each random draw of $\tilde{\alpha}_h$. Then, I will use those 4,000 samples of $\tilde{\alpha}_h$ and 4,000 samples of σ_y^2 as parameters, to obtain 4,000 random draws of \tilde{y}_k , with which I can create a histogram of my posterior predictive density.

```
# We first extract the posterior samples from our model using `as_draws` to extract 4,000 samples
# (1,000 per chain) for every parameter we're interested in.
```

```
samp <- as_draws_df(matri.fit)
# Check the size of our dataframe
dim(samp)
```

```
## [1] 4000 48
```

```
# Review names for later use
```

```
names(samp)[1:5]
```

```
## [1] "b_Intercept"          "sd_ethnicgroup__Intercept"
## [3] "sigma"                "r_ethnicgroup[1,Intercept]"
## [5] "r_ethnicgroup[2,Intercept]"
```

```

# Make a vector of the 4,000 samples for sigma_y (residuals)
sigmay_s <- samp$sigma
# Make a vector of the samples of mu_alpha
mualpha_s <- samp$b_Intercept
# Make a vector of the samples of sigma_alpha (sd of intercepts)
sigmaalpha_s <- samp$sd_ethnicgroup__Intercept
# Length of vectors (4,000)
S <- length(sigmay_s)
# Make sampling reproducible
set.seed(1234)
# Obtain a normally distributed random sample of alphas using the sampled posterior parameters
alphatilde_s <- rnorm(S, mualpha_s, sigmaalpha_s)
# Obtain a normally distributed random sample of ys using the sampled alphas and sigmay_s
ytilde_s <- rnorm(S, alphatilde_s, sigmay_s)
# Obtain point estimates and 95% CI (using tidybayes):
point_interval(ytilde_s, .point = mean)

```

```

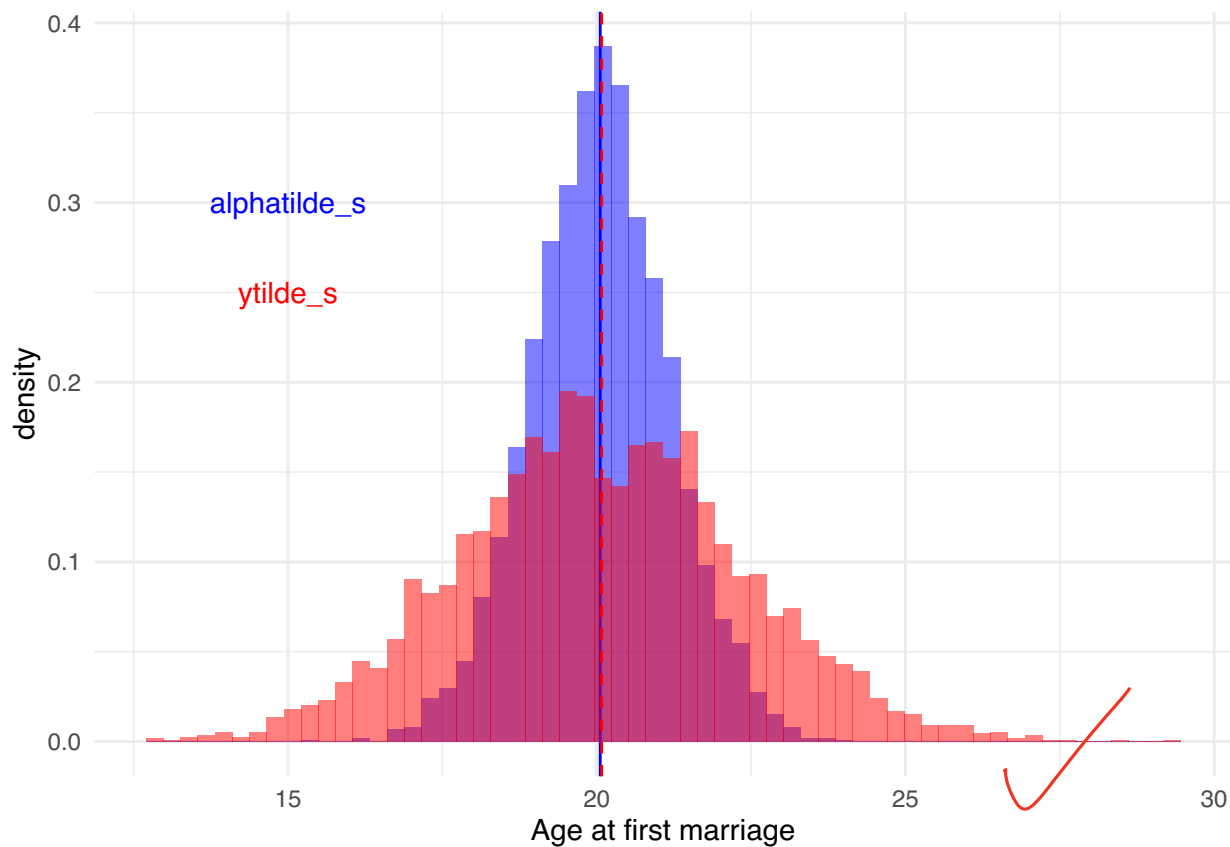
##           y      ymin      ymax .width .point .interval
## 1 20.08016 15.6973 24.54451 0.95 mean qi

```

```

# Visualize densities
p <- as_tibble(alphatilde_s) %>%
  ggplot(aes(alphatilde_s, after_stat(density), fill = "blue")) +
  geom_histogram(alpha = .5, fill = "blue", bins = 60) +
  theme_minimal() +
  xlab("Age at first marriage") +
  geom_vline(xintercept = mean(alphatilde_s), col = "blue")
q <- p + geom_histogram(as_tibble(ytilde_s), bins = 60,
  mapping = aes(ytilde_s, after_stat(density), fill = "red"),
  alpha = .5, fill = "red", size = 1.5) +
  geom_vline(xintercept = mean(ytilde_s), col = "red", linetype = "dashed")
q + annotate("text", x = 15, y = 0.3, label = "alphatilde_s", color = "blue") +
  annotate("text", x = 15, y = 0.25, label = "ytilde_s", color = "red")

```



- (b) Use the samples to construct a point prediction and 95% prediction interval for age at first marriage. In your answer, include the expression used for calculating the point prediction from the samples.

To calculate the point prediction from the samples, I start by using

```
# The point estimate is just the mean of the samples
(ytilde_s.mean <- mean(ytilde_s))
```

```
## [1] 20.08016
```

```
#For a simple approximation, I will just use the mean of sigmay_s to approximate the standard deviation
(sigmay_s.mean <- mean(sigmay_s))
```

```
## [1] 2.012571
```

```
# Credible intervals
(ytilde_s.ci <- c("lower" = ytilde_s.mean - (sigmay_s.mean*(qnorm(0.975))), "upper" = ytilde_s.mean + (sigmay_s.mean*(qnorm(0.975)))))
```

```
##      lower      upper
## 16.13559 24.02473
```

A more exact approximation can be done using tidybayes:

```
# Obtain point estimates and 95% CI (using tidybayes):
point_interval(ytilde_s, .width = 0.95, .point = mean)
```

```
##           y      ymin      ymax .width .point .interval
## 1 20.08016 15.6973 24.54451   0.95   mean          qi
```

- (c) What is the probability that the observed age at first marriage will be greater than \bar{y} ? In your answer, include the expression used for calculating this probability from the samples.


The probability that the observed age at first marriage will be greater than \bar{y} is given by:

$$P(\tilde{y}_k > \bar{y}) = P(\tilde{y}_k - \tilde{\mu}_\alpha > 0)$$

For this, I can subtract the samples of $\tilde{\mu}_\alpha$ from \tilde{y}_k and find the percentage greater than \bar{y} :

```
q4.c <- ytilde_s - mualpha_s
n <- length(ytilde_s)
length(q4.c[q4.c>0])/n
```

```
## [1] 0.49575
```



So, the probability that the observed age at first marriage will be greater than \bar{y} according to our sampled distribution is 0.496, or almost 50%.

Exercise 5 (extra credit, 3 points)

Obtain the joint distribution of $(y_1, y_2) | \mu_\alpha, \sigma_\alpha^2, \sigma_y^2$ in the setting where $j[1] \neq j[2]$ and in the setting where $j[1] = j[2]$.

Hints:

- First consider the univariate distribution of $y_i | \mu_\alpha, \sigma_\alpha^2, \sigma_y^2$.
- You may find it helpful to write y_i as the sum of parameters (thus random variables) μ_α , a parameter to capture the variability across counties, and a parameter to capture variability across observations.