50/40

Applied Bayesian modeling - HW4, part 1

Álvaro J. Castro Rivadeneira

November 2, 2022

In this HW, we are going to analyze the wells data (briefly mentioned in class) using logistic regression models, and do model checking. HW4-part1 is based on module 11, part 1 (in-sample checking). Later parts of this analysis include approximate leave-one-out validation and testing sensitivity of results to choice of priors.

If you'd like a refresher on logistic regression, and want to read about it in a Bayesian context, you may find these texts helpful (do add others you recommend on the slack!):

- https://bookdown.org/marklhc/notes_bookdown/generalized-linear-models.html#binary-logistic-regression
- https://www.bayesrulesbook.com/chapter-13.html

For model fitting, you can choose if you want to fit the models using the brms or rstan package functions (or both!). Either way, you will need to investigate how to fit a logistic regression model. Consider using help functions (i.e. check out the family option in brm), consider the resources, and/or do a google search for vignettes or tutorials to do logistic regression with brm or stan.

Choice of priors will be discussed further in part 2. A default recommendation (based on centered covariates) varies across references but generally, distributions with fatter tails (as compared to normal densities) are recommended, such as a t-distribution. For part 1 of the HW, when using brm, you may use brm-default priors (based on centered covariates, the default here is to use a student_t(3, 0, 2.5) for the intercept, flat priors are used for other coefficients). When using stan, you may use the same priors, or consider a student_t(df = 7, location = 0, scale = 2.5), as recommended here https://avehtari.github.io/modelselection/diabetes.html.

For all model fits, include centered covariates (i.e. subtract the mean of the covariate) and make sure Rhat and effectve sample sizes don't suggest any issues.

Wells data

Information taken from https://cran.r-project.org/web/packages/rstanarm/vignettes/binomial.html. The data are described here https://vincentarelbundock.github.io/Rdatasets/doc/carData/Wells.html

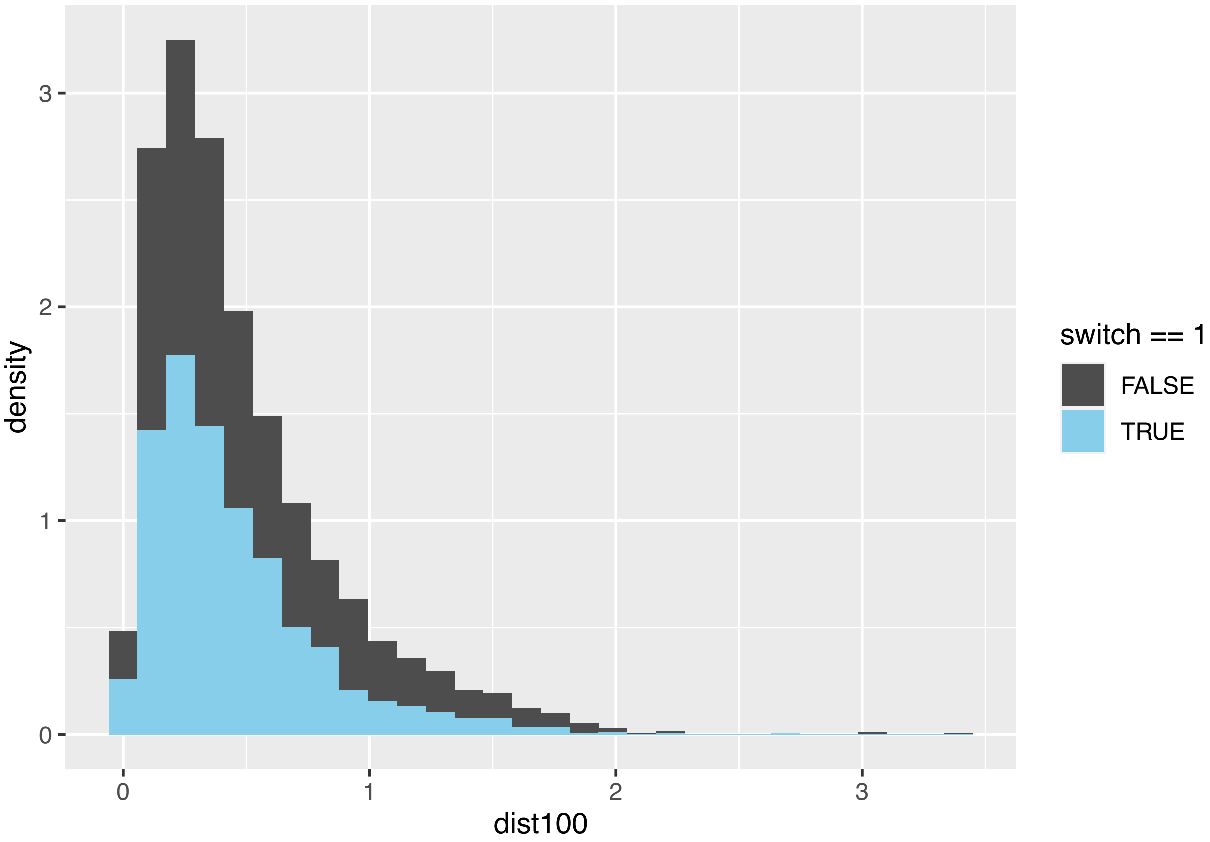
Gelman and Hill describe a survey of 3200 residents in a small area of Bangladesh suffering from arsenic contamination of groundwater. Respondents with elevated arsenic levels in their wells had been encouraged to switch their water source to a safe public or private well in the nearby area and the survey was conducted several years later to learn which of the affected residents had switched wells. The goal of the analysis presented by Gelman and Hill is to learn about the factors associated with switching wells.

Reading in the data and creating some transformed variables:

```r
url <- "http://stat.columbia.edu/~gelman/arm/examples/arsenic/wells.dat"
wells <- read.table(url)
wells <- wells %>%
  # adding some transformed and centered variables
  mutate(y = switch,
         dist100 = dist / 100,
         # rescale the dist variable (measured in meters) so that it is measured in units of 100 meters
        c_dist100 = dist100 - mean (dist100),
        c_arsenic = arsenic - mean (arsenic))
```

A simple plot: blue bars correspond to the 1737 residents who said they switched wells and darker bars show the distribution of dist100 for the 1283 residents who didn't switch. As we would expect, for the residents who switched wells, the distribution of dist100 is more concentrated at smaller distances.

```r
ggplot(wells, aes(x = dist100, y = ..density.., fill = switch == 1)) +
  geom_histogram() +
  scale_fill_manual(values = c("gray30", "skyblue"))
```



Question 1: fitting a logistic regression model (warm-up exercise) 10

Fit the following simple logistic regression model:

$$\begin{aligned} y_i &\sim Bern(\theta_i), \\ logit(\theta_i) &= \beta_0 + \beta_1 \cdot (d_i - \bar{d}), \end{aligned}$$

where $y_i = 1$ if household i switched wells , 0 otherwise (recorded by the variable `switch` in the dataset), θ_i refers to its probability of switching and d_i to its distance to the nearest safe well (measured in 100 meters, `dist100` in the well dataset).

Report point estimates and 95% CIs for β_0 and β_1. Interpret these estimates in terms of odds ratios.

Answer

First, I like to get a sense of the distribution of the data:

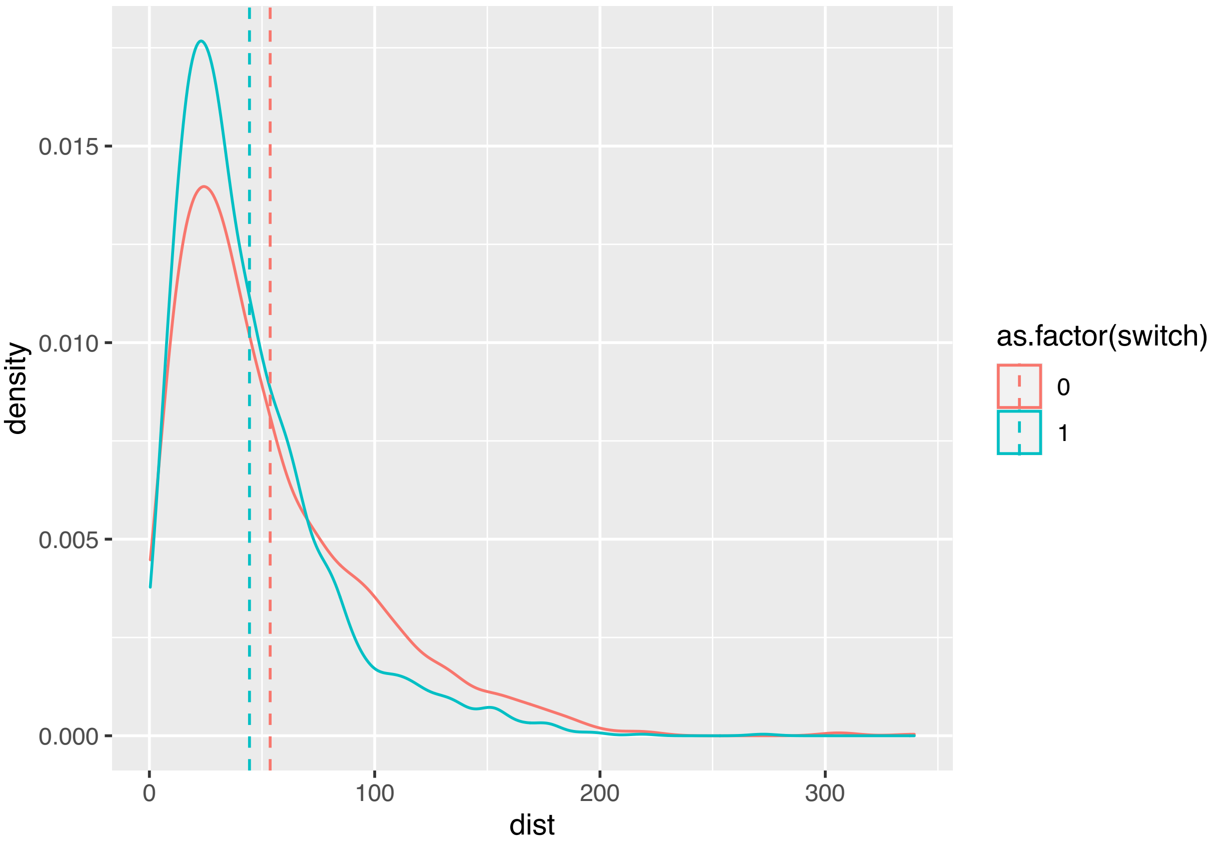
```
# First a look at all the data
summary(wells)
```

```
##      switch          arsenic           dist             assoc       
##  Min.   :0.0000   Min.   :0.510   Min.   :  0.387   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:0.820   1st Qu.: 21.117   1st Qu.:0.0000  
##  Median :1.0000   Median :1.300   Median : 36.761   Median :0.0000  
##  Mean   :0.5752   Mean   :1.657   Mean   : 48.332   Mean   :0.4228  
##  3rd Qu.:1.0000   3rd Qu.:2.200   3rd Qu.: 64.041   3rd Qu.:1.0000  
##  Max.   :1.0000   Max.   :9.650   Max.   :339.531   Max.   :1.0000  
##       educ              y             dist100          c_dist100      
##  Min.   : 0.000   Min.   :0.0000   Min.   :0.00387   Min.   :-0.4794  
##  1st Qu.: 0.000   1st Qu.:0.0000   1st Qu.:0.21117   1st Qu.:-0.2721  
##  Median : 5.000   Median :1.0000   Median :0.36762   Median :-0.1157  
##  Mean   : 4.828   Mean   :0.5752   Mean   :0.48332   Mean   : 0.0000  
##  3rd Qu.: 8.000   3rd Qu.:1.0000   3rd Qu.:0.64041   3rd Qu.: 0.1571  
##  Max.   :17.000   Max.   :1.0000   Max.   :3.39531   Max.   : 2.9120  
##    c_arsenic      
##  Min.   :-1.1469  
##  1st Qu.:-0.8369  
##  Median :-0.3569  
##  Mean   : 0.0000  
##  3rd Qu.: 0.5431  
##  Max.   : 7.9931  
```

```
# Now, a different, simple plot to look at differences in switch
mus <- ddply(wells, "switch", summarise, grp.mean=mean(dist))
mus
```

```
##   switch grp.mean
## 1      0 53.61148
## 2      1 44.43218
```

```
ggplot(wells, aes(x=dist, color=as.factor(switch))) +
  geom_density() +
  geom_vline(data=mus, aes(xintercept=grp.mean, color=as.factor(switch)),
             linetype="dashed")
```



```
# Finally, I like to run a traditional glm to compare results:
hw4q.glm.fit_1 <- glm(y ~ 1 + c_dist100, data = wells, family = "binomial")
summary(hw4q.glm.fit_1)
```

```
##
## Call:
## glm(formula = y ~ 1 + c_dist100, family = "binomial", data = wells)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4406  -1.3058   0.9669   1.0308   1.6603
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.30539    0.03707   8.237  < 2e-16 ***
## c_dist100   -0.62188    0.09743  -6.383 1.74e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 4076.2  on 3018  degrees of freedom
## AIC: 4080.2
##
## Number of Fisher Scoring iterations: 4
```

```r
cbind(Estimate = coef(hw4q.glm.fit_1), confint(hw4q.glm.fit_1))
```

```
##               Estimate      2.5 %     97.5 %
## (Intercept)  0.3053922  0.2328550  0.3781997
## c_dist100   -0.6218819 -0.8140762 -0.4319795
```

```r
exp(cbind(OR = coef(hw4q.glm.fit_1), confint(hw4q.glm.fit_1)))
```

```
##                   OR     2.5 %    97.5 %
## (Intercept) 1.357157 1.2621985 1.4596543
## c_dist100   0.536933 0.4430484 0.6492227
```

```r
# Check non-centered results for comparison
hw4q.glm.fit_2 <- glm(y ~ 1 + dist100, data = wells, family = "binomial")
summary(hw4q.glm.fit_2)
```

```
## 
## Call:
## glm(formula = y ~ 1 + dist100, family = "binomial", data = wells)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -1.4406  -1.3058   0.9669   1.0308   1.6603  
## 
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)    
## (Intercept)  0.60596    0.06031  10.047  < 2e-16 ***
## dist100     -0.62188    0.09743  -6.383 1.74e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 4076.2  on 3018  degrees of freedom
## AIC: 4080.2
## 
## Number of Fisher Scoring iterations: 4
```

```r
cbind(Estimate = coef(hw4q.glm.fit_2), confint(hw4q.glm.fit_2))
```

```
##               Estimate      2.5 %     97.5 %
## (Intercept)  0.6059594  0.4882230  0.7246814
## dist100     -0.6218819 -0.8140762 -0.4319795
```

```r
exp(cbind(OR = coef(hw4q.glm.fit_2), confint(hw4q.glm.fit_2)))
```

```
##                   OR     2.5 %    97.5 %
## (Intercept) 1.833010 1.6294182 2.0640733
## dist100     0.536933 0.4430484 0.6492227
```

```r
# Find OR for d = d_bar using the second fit (should = first intercept OR)
dist100_bar <- mean(wells$dist100)
beta_0 <- coef(hw4q.glm.fit_2)[1]
beta_1 <- coef(hw4q.glm.fit_2)[2]
(or_d_bar <- exp(beta_0 + (dist100_bar * beta_1)))
```

```
## (Intercept)
##    1.357157
```

```r
# The results are what was expected!
```

Now to fit a model with one predictor:

```r
t_prior <- set_prior("student_t(7, 0, 2.5)", class = "Intercept")

hw4q1.fit <- brm(formula = y ~ 1 + c_dist100,
                 file = "output/hw4q1",
                 data = wells, sample_prior = T,
                 prior = c(t_prior),
                 family = bernoulli(link = "logit"),
                 chains = 4, iter = 2000, warmup = 1000,
                 cores = getOption("mc.cores", 4),
                 thin = 1, seed = 1234)
```

✓

Now, the point estimates and 95% CIs for β_0 and β_1:

```r
# Reviewing output and diagnostics
summary(hw4q1.fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ 1 + c_dist100
##    Data: wells (Number of observations: 3020)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.31      0.04     0.23     0.37 1.00     3255     2505
## c_dist100    -0.62      0.10    -0.81    -0.44 1.00     3433     2185
##
## 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).
```

✓

```r
summary(hw4q1.fit)$fixed
```

```
##             Estimate  Est.Error   l-95% CI   u-95% CI     Rhat Bulk_ESS
## Intercept  0.3050530 0.03647829  0.2328800  0.3745385 1.002399  3255.29
## c_dist100 -0.6240421 0.09683955 -0.8113021 -0.4360378 1.001418  3432.76
##           Tail_ESS
## Intercept 2505.078
## c_dist100 2184.704
```

```
# Intercept and coefficients
fixef(hw4q1.fit)
```

```
##             Estimate  Est.Error       Q2.5      Q97.5
## Intercept  0.3050530 0.03647829  0.2328800  0.3745385
## c_dist100 -0.6240421 0.09683955 -0.8113021 -0.4360378
```

```
(beta_0 <- fixef(hw4q1.fit)[1])
```

```
## [1] 0.305053
```

✓

```
(or_zerodist <- exp(beta_0))
```

```
## [1] 1.356697
```

✓

```
(p_zerodist <- exp(beta_0) / (1 + exp(beta_0)))
```

```
## [1] 0.5756773
```

```
# Check results using plogis
plogis(fixef(hw4q1.fit)[1])
```

```
## [1] 0.5756773
```

```
# Results match what was expected
```

The 95% CI for β_0 is 0.233 - 0.375 and the 95% CI for β_1 is -0.811 - -0.436. To interpret these results we need to exponentiate them:

```
exp(fixef(hw4q1.fit)[,-2])
```

```
##            Estimate      Q2.5     Q97.5
## Intercept 1.3566969 1.2622300 1.4543200
## c_dist100 0.5357744 0.4442792 0.6465933
```

✓

It is worth noting these results are almost identical to what I obtained with a traditional frequentist logistic regression.

The interpretation for β_0 in terms of the odds ratio is that participants who live at exactly the mean distance from the closest known safe well (so that $(d_i - \bar{d}) = 0$) have 1.36 times the odds (or a 57.6% probability) of switching to another well from an unsafe well (95% CI for OR: 1.26 - 1.45). On the other hand, for β_1 we can say that for every 100 meters away from the mean distance to to the closest known safe well ($(d_i - \bar{d}) = 1$), the odds of switching to another well from an unsafe well is 0.54 (95% CI for OR: 0.44 - 0.65). Stated differently, for every 100 meters further away from the closest known safe well, they have a 46% decrease in the odds of making a switch to another well from an unsafe well.

✓

Question 2: Models with distance and arsenic

10

Now consider models that include a second predictor, which is the arsenic level in the respondents' well (called `arsenic` in the dataset). Fit model (2), which has distance/100 and arsenic levels as predictors, as well as model (3), which has both predictors and their interaction term.

Write out the equations for both models, and construct one plot that shows the relation between the estimated switch probability and arsenic levels for both models for households that are 100 meters away from a safe well (use posterior means of the regression coefficients and show the model with the interaction term in a dashed red line). Interpret the difference between the fitted regression lines.

Answer

The probability θ_i of making a switch y_i for both models is given by:

$$y_i \sim Bern(\theta_i),$$

Model (2) would have the following equation:

$$logit(\theta_i) = \beta_0 + \beta_1 \cdot (d_i - \bar{d}) + \beta_2 \cdot (a_i - \bar{a}),$$

Model (3) would have the following equation:

$$logit(\theta_i) = \beta_0 + \beta_1 \cdot (d_i - \bar{d}) + \beta_2 \cdot (a_i - \bar{a}) + \beta_3 \cdot (d_i - \bar{d}) \cdot (a_i - \bar{a}),$$

Now, to fit model (2):

```
hw4q2_2.fit <- brm(formula = y ~ 1 + c_dist100 + c_arsenic,
                   file = "output/hw4q2_2",
                   data = wells, sample_prior = T,
                   prior = c(t_prior),
                   family = bernoulli(link = "logit"),
                   chains = 4, iter = 2000, warmup = 1000,
                   cores = getOption("mc.cores", 4),
                   thin = 1, seed = 1234)
```

✓

. . . and a fit for model (3):

```
hw4q2_3.fit <- brm(formula = y ~ 1 + c_dist100 * c_arsenic,
                   file = "output/hw4q2_3",
                   data = wells, sample_prior = T,
                   prior = c(t_prior),
                   family = bernoulli(link = "logit"),
                   chains = 4, iter = 2000, warmup = 1000,
                   cores = getOption("mc.cores", 4),
                   thin = 1, seed = 1234)
```

✓

Now to look at and compare the results from both models:

```
# Reviewing output and diagnostics
summary(hw4q2_2.fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ 1 + c_dist100 + c_arsenic
##    Data: wells (Number of observations: 3020)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.33      0.04     0.26     0.41 1.00     3710     3140
## c_dist100    -0.90      0.10    -1.10    -0.70 1.00     3153     2709
## c_arsenic     0.46      0.04     0.39     0.54 1.00     3446     2756
##
## 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).
```

```
summary(hw4q2_3.fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ 1 + c_dist100 * c_arsenic
##    Data: wells (Number of observations: 3020)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##                     Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept               0.35      0.04     0.28     0.43 1.00     3925     2946
## c_dist100              -0.88      0.10    -1.08    -0.68 1.00     3722     3345
## c_arsenic               0.47      0.04     0.39     0.56 1.00     3771     3412
## c_dist100:c_arsenic    -0.18      0.10    -0.38     0.02 1.00     4155     3316
##
## 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).
```

```
# Fixed effects
summary(hw4q2_2.fit)$fixed
```

```
##             Estimate  Est.Error   l-95% CI   u-95% CI      Rhat Bulk_ESS
## Intercept  0.3339635 0.03867579  0.2579909  0.4081779 0.9998892 3709.722
## c_dist100 -0.9009966 0.10440376 -1.1029546 -0.6959655 1.0009093 3153.304
## c_arsenic  0.4632278 0.03979575  0.3882866  0.5437699 1.0000655 3446.157
##           Tail_ESS
## Intercept 3140.474
## c_dist100 2709.029
## c_arsenic 2755.749
```

```
summary(hw4q2_3.fit)$fixed
```

```
##                       Estimate  Est.Error   l-95% CI    u-95% CI     Rhat
## Intercept            0.3514297 0.03902928  0.2753226  0.42896865 1.000688
## c_dist100           -0.8759479 0.10454521 -1.0810551 -0.67511025 1.000264
## c_arsenic            0.4710138 0.04216288  0.3876526  0.55563532 1.000302
## c_dist100:c_arsenic -0.1826818 0.10137007 -0.3811156  0.01628972 1.000623
##                      Bulk_ESS Tail_ESS
## Intercept            3925.365 2946.161
## c_dist100            3722.054 3345.035
## c_arsenic            3771.288 3412.002
## c_dist100:c_arsenic 4154.519 3316.134
```

```
# Intercept and coefficients
fixef(hw4q2_2.fit)
```

```
##             Estimate  Est.Error       Q2.5      Q97.5
## Intercept  0.3339635 0.03867579  0.2579909  0.4081779
## c_dist100 -0.9009966 0.10440376 -1.1029546 -0.6959655
## c_arsenic  0.4632278 0.03979575  0.3882866  0.5437699
```

```
fixef(hw4q2_3.fit)
```

```
##                       Estimate  Est.Error       Q2.5       Q97.5
## Intercept            0.3514297 0.03902928  0.2753226  0.42896865
## c_dist100           -0.8759479 0.10454521 -1.0810551 -0.67511025
## c_arsenic            0.4710138 0.04216288  0.3876526  0.55563532
## c_dist100:c_arsenic -0.1826818 0.10137007 -0.3811156  0.01628972
```

```
exp(fixef(hw4q2_2.fit)[,-2])
```

```
##            Estimate     Q2.5     Q97.5
## Intercept 1.3964922 1.294327 1.5040747
## c_dist100 0.4061647 0.331889 0.4985928
## c_arsenic 1.5891954 1.474452 1.7224882
```

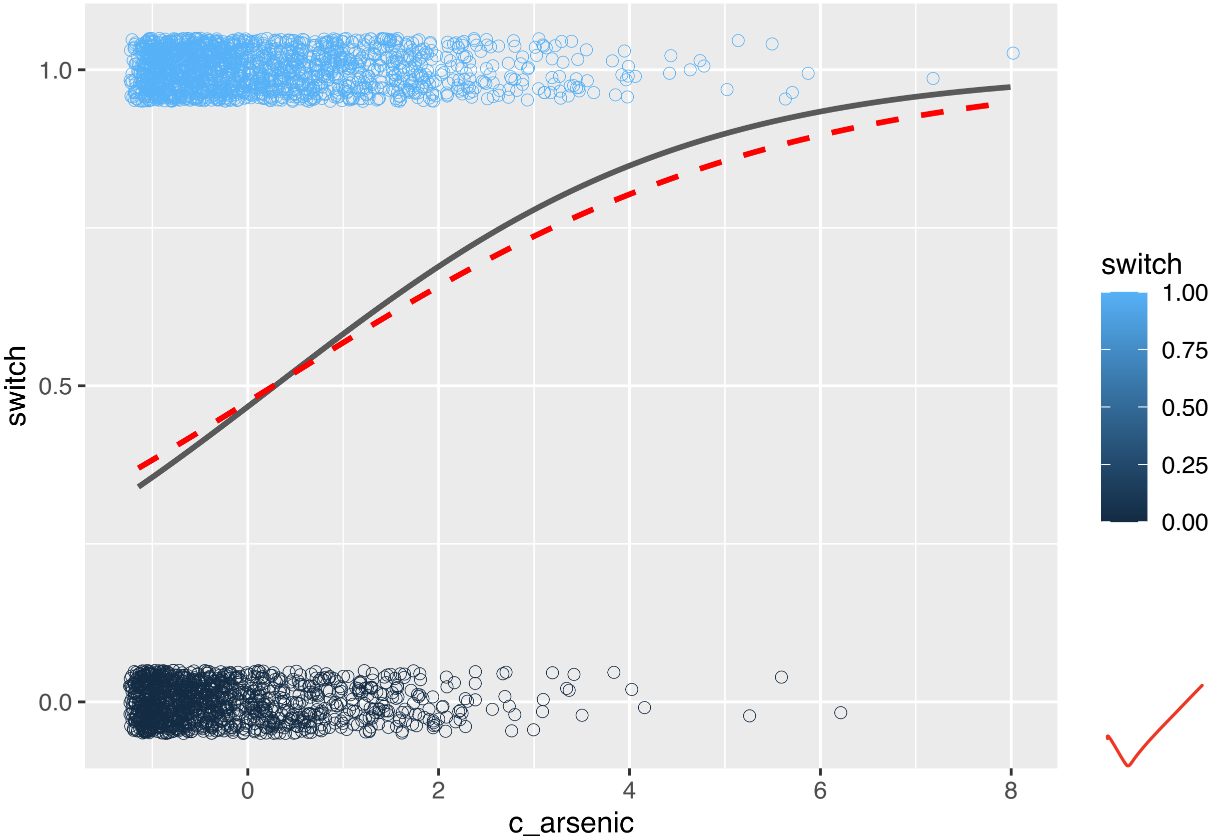
```
exp(fixef(hw4q2_3.fit)[,-2])
```

```
##                      Estimate      Q2.5     Q97.5
## Intercept           1.4210979 1.3169555 1.5356729
## c_dist100           0.4164671 0.3392374 0.5091003
## c_arsenic           1.6016170 1.4735178 1.7430480
## c_dist100:c_arsenic 0.8330332 0.6830989 1.0164231
```

Finally, a plot that shows the relation between the estimated switch probability and arsenic levels for both models for households that are 100 meters away from a safe well. Basing myself on the `rstanarm` vignette:

```
# Predicted probability as a function of x
pr_switch_2 <- function(x, y, ests) plogis(ests[1] + ests[2] * x + ests[3] * y)
pr_switch_3 <- function(x, y, ests) plogis(ests[1] + ests[2] * x + ests[3] * y + (ests[4] * x * y))
# A function to slightly jitter the binary data
jitt <- function(...) {
  geom_point(aes_string(...), position = position_jitter(height = 0.05, width = 0.1),
             size = 2, shape = 21, stroke = 0.2)
}
# Set the centered distance value as 100 meters away from a safe well
```

```
d.is.100 <- (1 - dist100_bar)
# The plot
ggplot(wells, aes(x = c_arsenic, y = switch, color = switch)) +
  scale_y_continuous(breaks = c(0, 0.5, 1)) +
  jitt(x="c_arsenic") +
  stat_function(fun = pr_switch_2, args = list(x = d.is.100, ests = fixef(hw4q2_2.fit)),
                size = 1, color = "gray35") +
  stat_function(fun = pr_switch_3, args = list(x = d.is.100, ests = fixef(hw4q2_3.fit)),
                size = 1, color = "red", linetype = "dashed")
```



✓

```
# Check to see plotted results match what we expect when c_arsenic = 4
c_As.is.3 <- 3
# First, for model (2), use estimates obtained previously (should be above 0.75):
(p2.As.is.3 <- plogis(0.3339635 + (-0.9009966 * d.is.100) + (0.4632278 * c_As.is.3)))
```

```
## [1] 0.7787021
```

```
# Answer: 0.7787021
# First, for model (3), use estimates obtained previously (should be slightly below 0.75):
(p3.As.is.3 <- plogis(0.3514297 + (-0.8759479 * d.is.100) + (0.4710138 * c_As.is.3) + (-0.1826818 * d.i
```

```
## [1] 0.7366677
```

```
# Answer: 0.7366677
# The results seem reasonable and correct.
```

I want to check my work and make sure I'm doing this correctly:

```
# First I will run a traditional GLM to see if I get similar results
# Model (2)
hw4q2_2.glm.fit_1 <- glm(y ~ 1 + c_dist100 + c_arsenic, data = wells, family = "binomial")
summary(hw4q2_2.glm.fit_1)
```

```
##
## Call:
## glm(formula = y ~ 1 + c_dist100 + c_arsenic, family = "binomial",
##     data = wells)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6351  -1.2139   0.7786   1.0702   1.7085
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.33286    0.03832   8.687   <2e-16 ***
## c_dist100   -0.89664    0.10435  -8.593   <2e-16 ***
## c_arsenic    0.46077    0.04138  11.134   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3930.7  on 3017  degrees of freedom
## AIC: 3936.7
##
## Number of Fisher Scoring iterations: 4
```

```
cbind(Estimate = coef(hw4q2_2.glm.fit_1), confint(hw4q2_2.glm.fit_1))
```

```
##               Estimate      2.5 %     97.5 %
## (Intercept)  0.3328559  0.2579563  0.4081723
## c_dist100   -0.8966442 -1.1029187 -0.6937223
## c_arsenic    0.4607749  0.3808083  0.5430701
```

```
exp(cbind(OR = coef(hw4q2_2.glm.fit_1), confint(hw4q2_2.glm.fit_1)))
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 1.3949463 1.2942823 1.5040663
## c_dist100   0.4079363 0.3319009 0.4997125
## c_arsenic   1.5853020 1.4634670 1.7212833
```

```
# Very similar results
## Not centered Model (2)
hw4q2_2.glm.fit_2 <- glm(y ~ 1 + dist100 + arsenic, data = wells, family = "binomial")
summary(hw4q2_2.glm.fit_2)
```

```
##
## Call:
## glm(formula = y ~ 1 + dist100 + arsenic, family = "binomial",
##     data = wells)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6351  -1.2139   0.7786   1.0702   1.7085
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.002749   0.079448   0.035    0.972
## dist100     -0.896644   0.104347  -8.593   <2e-16 ***
## arsenic      0.460775   0.041385  11.134   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3930.7  on 3017  degrees of freedom
## AIC: 3936.7
##
## Number of Fisher Scoring iterations: 4
```

```
cbind(Estimate = coef(hw4q2_2.glm.fit_2), confint(hw4q2_2.glm.fit_2))
```

```
##                 Estimate      2.5 %     97.5 %
## (Intercept)  0.002748671 -0.1531226  0.1583687
## dist100     -0.896644172 -1.1029187 -0.6937223
## arsenic      0.460774949  0.3808083  0.5430701
```

```
exp(cbind(OR = coef(hw4q2_2.glm.fit_2), confint(hw4q2_2.glm.fit_2)))
```

```
##                    OR     2.5 %    97.5 %
## (Intercept) 1.0027525 0.8580245 1.1715981
## dist100     0.4079363 0.3319009 0.4997125
## arsenic     1.5853020 1.4634670 1.7212833
```

```
# Using non-centered data I get a different intercept, but the same coefficients for variables
# Model  (3)
hw4q2_3.glm.fit_1 <- glm(y ~ 1 + c_dist100 * c_arsenic, data = wells, family = "binomial")
summary(hw4q2_3.glm.fit_1)
```

```
##
## Call:
```

```
## glm(formula = y ~ 1 + c_dist100 * c_arsenic, family = "binomial",
##     data = wells)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7823  -1.2004   0.7696   1.0816   1.8476
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.35109    0.03985   8.810   <2e-16 ***
## c_dist100           -0.87365    0.10480  -8.337   <2e-16 ***
## c_arsenic            0.46951    0.04207  11.159   <2e-16 ***
## c_dist100:c_arsenic -0.17891    0.10233  -1.748   0.0804 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3927.6  on 3016  degrees of freedom
## AIC: 3935.6
##
## Number of Fisher Scoring iterations: 4
```

```
cbind(Estimate = coef(hw4q2_3.glm.fit_1), confint(hw4q2_3.glm.fit_1))
```

```
##                       Estimate      2.5 %      97.5 %
## (Intercept)          0.3510941  0.2732603  0.42951145
## c_dist100           -0.8736527 -1.0808894 -0.66992679
## c_arsenic            0.4695081  0.3882846  0.55325092
## c_dist100:c_arsenic -0.1789060 -0.3795821  0.02232171
```

```
exp(cbind(OR = coef(hw4q2_3.glm.fit_1), confint(hw4q2_3.glm.fit_1)))
```

```
##                            OR     2.5 %   97.5 %
## (Intercept)         1.4206210 1.3142423 1.536507
## c_dist100           0.4174240 0.3392936 0.511746
## c_arsenic           1.5992074 1.4744494 1.738897
## c_dist100:c_arsenic 0.8361845 0.6841473 1.022573
```

```
# Very similar results
# Not centered Model (3)
hw4q2_3.glm.fit_2 <- glm(y ~ 1 + dist100 * arsenic, data = wells, family = "binomial")
summary(hw4q2_3.glm.fit_2)
```

```
##
## Call:
## glm(formula = y ~ 1 + dist100 * arsenic, family = "binomial",
##     data = wells)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7823  -1.2004   0.7696   1.0816   1.8476
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -0.14787    0.11754  -1.258  0.20838
## dist100         -0.57722    0.20918  -2.759  0.00579 **
## arsenic          0.55598    0.06932   8.021 1.05e-15 ***
## dist100:arsenic -0.17891    0.10233  -1.748  0.08040 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4118.1  on 3019  degrees of freedom
## Residual deviance: 3927.6  on 3016  degrees of freedom
## AIC: 3935.6
##
## Number of Fisher Scoring iterations: 4
```

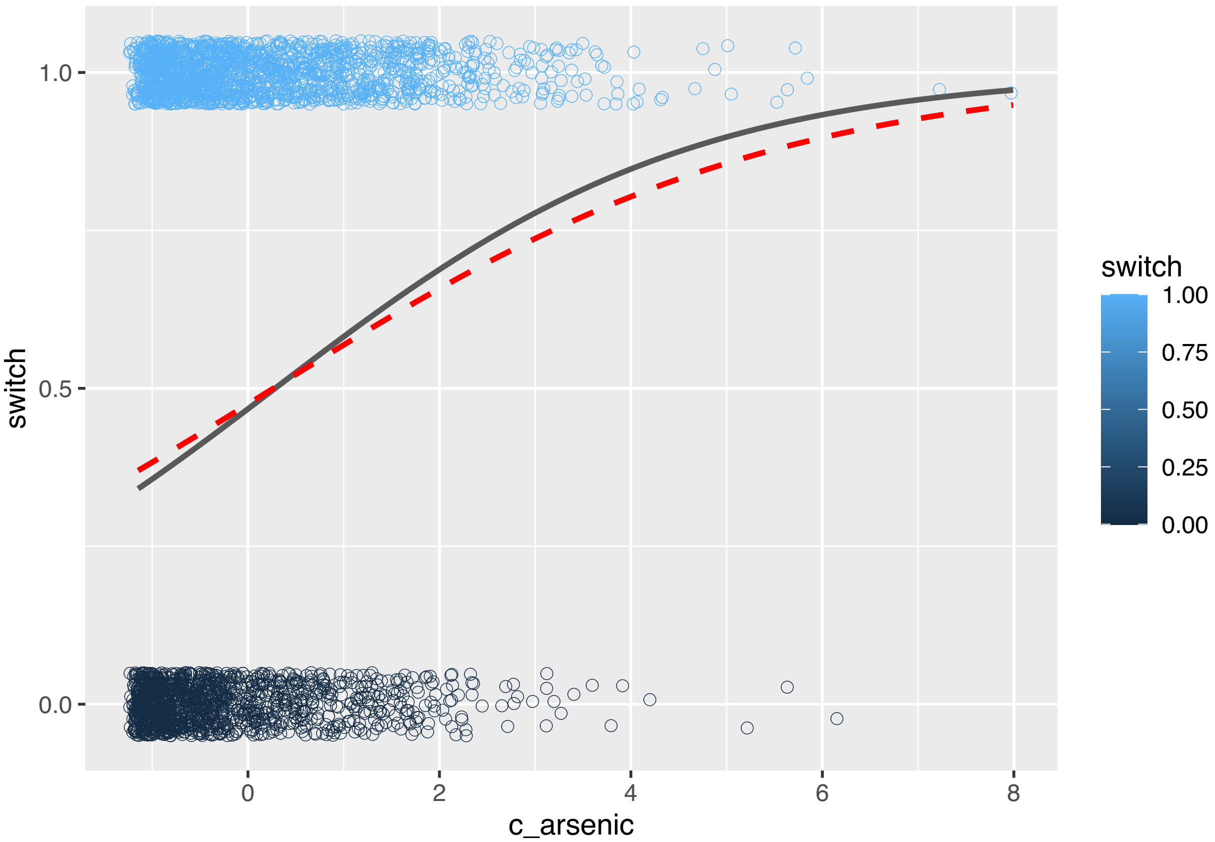
```
cbind(Estimate = coef(hw4q2_3.glm.fit_2), confint(hw4q2_3.glm.fit_2))
```

```
##                   Estimate      2.5 %      97.5 %
## (Intercept)     -0.1478681 -0.3787334  0.08216010
## dist100         -0.5772178 -0.9907998 -0.17009552
## arsenic          0.5559767  0.4219567  0.69380626
## dist100:arsenic -0.1789060 -0.3795821  0.02232171
```

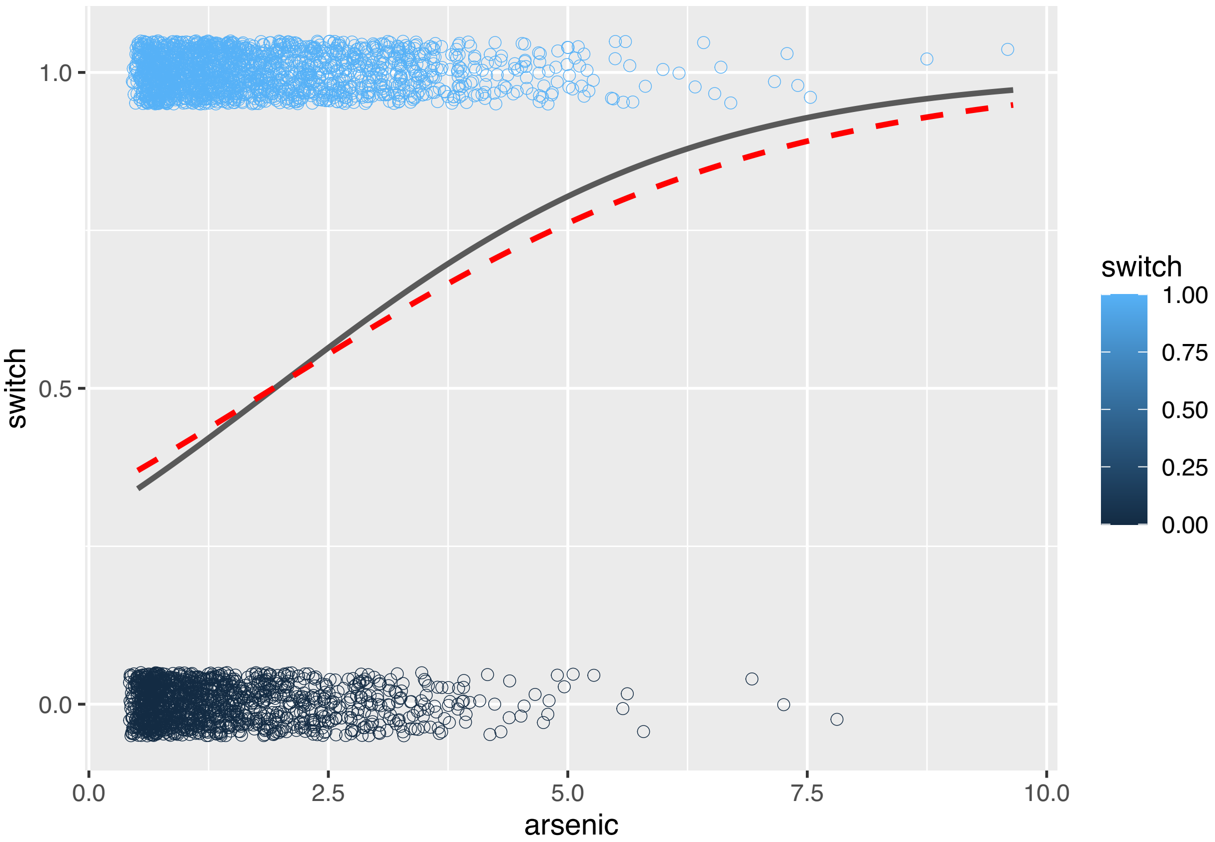
```
exp(cbind(OR = coef(hw4q2_3.glm.fit_2), confint(hw4q2_3.glm.fit_2)))
```

```
##                        OR     2.5 %    97.5 %
## (Intercept)     0.8625449 0.6847281 1.0856296
## dist100         0.5614583 0.3712796 0.8435842
## arsenic         1.7436432 1.5249425 2.0013186
## dist100:arsenic 0.8361845 0.6841473 1.0225727
```

```
# Using non-centered results, only the interaction coefficient is the same in both models
# Now a plot using the centered coefficients
ggplot(wells, aes(x = c_arsenic, y = switch, color = switch)) +
  scale_y_continuous(breaks = c(0, 0.5, 1)) +
  jitt(x="c_arsenic") +
  stat_function(fun = pr_switch_2, args = list(x = d.is.100, ests = coef(hw4q2_2.glm.fit_1)),
                size = 1, color = "gray35") +
  stat_function(fun = pr_switch_3, args = list(x = d.is.100, ests = coef(hw4q2_3.glm.fit_1)),
                size = 1, color = "red", linetype = "dashed")
```



```r
# Now a plot using the non-centered coefficients
ggplot(wells, aes(x = arsenic, y = switch, color = switch)) +
  scale_y_continuous(breaks = c(0, 0.5, 1)) +
  jitt(x="arsenic") +
  stat_function(fun = pr_switch_2, args = list(x = 1, ests = coef(hw4q2_2.glm.fit_2)),
                size = 1, color = "gray35") +
  stat_function(fun = pr_switch_3, args = list(x = 1, ests = coef(hw4q2_3.glm.fit_2)),
                size = 1, color = "red", linetype = "dashed")
```



```
# They look exactly the same. I am fairly confident in my results
```

Interpretation: In both models, increasing concentrations of arsenic lead to increasing probability of switching to another well from an unsafe well. In fact, the coefficient for `c_arsenic`, β_2 is essentially identical in both models (0.46 and 0.47). This coefficient means that an increase in arsenic is associated with an increased possibility of switching wells. Nonetheless, the interaction term between arsenic and distance has the opposite effect,as it has a negative coefficient (-0.18). When multiplied by $(d_i - \bar{d}) = 0.52$, and then exponentiated, it equals 0.91. This means that it has the effect of decreasing the slope, and thus, reducing odds of switching wells as the arsenic concentration increases. The effect is rather small, but it is evident in the red dashed line which starts higher than Model (2), but then reduces as the concentration of arsenic increases. In other words, Model(3) reduces the magnitude of change of the probability of switching wells as the concentration of arsenic increases.

Question 3: Residual plots

10

Produce residual plots for model 3, to show how residuals in that model vary with distance and arsenic. Start by calculating the residuals as discussed in module 11.

Answer
Following the examples given in the lecture and module 11, first we calculate the residuals:

```
ynew_si <- posterior_predict(hw4q2_3.fit) # adding si to indicate the dimension used
dim(ynew_si)
```

```
## [1] 4000 3020
```

```
ytildehat_i <- apply(ynew_si, 2, mean)
length(ytildehat_i)
```

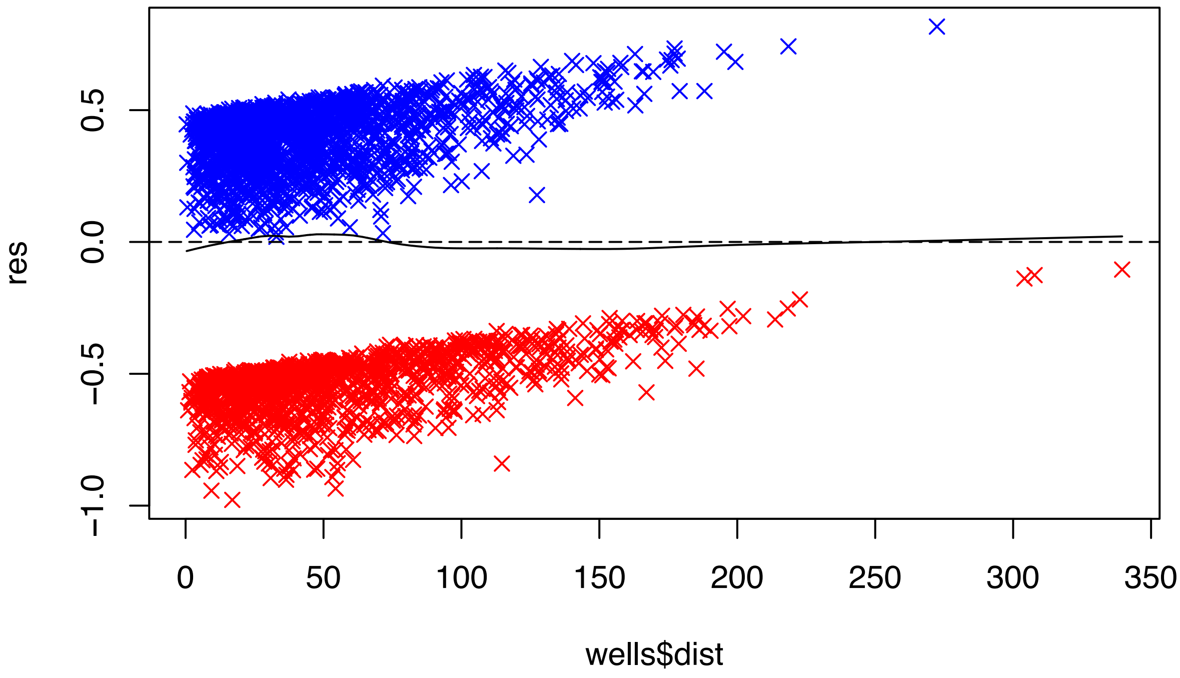
```
## [1] 3020
```

```
res <- wells$y - ytildehat_i
summary(res)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.9782500 -0.5117500  0.2561250  0.0000664  0.4427500  0.8167500
```

Now for some plots:

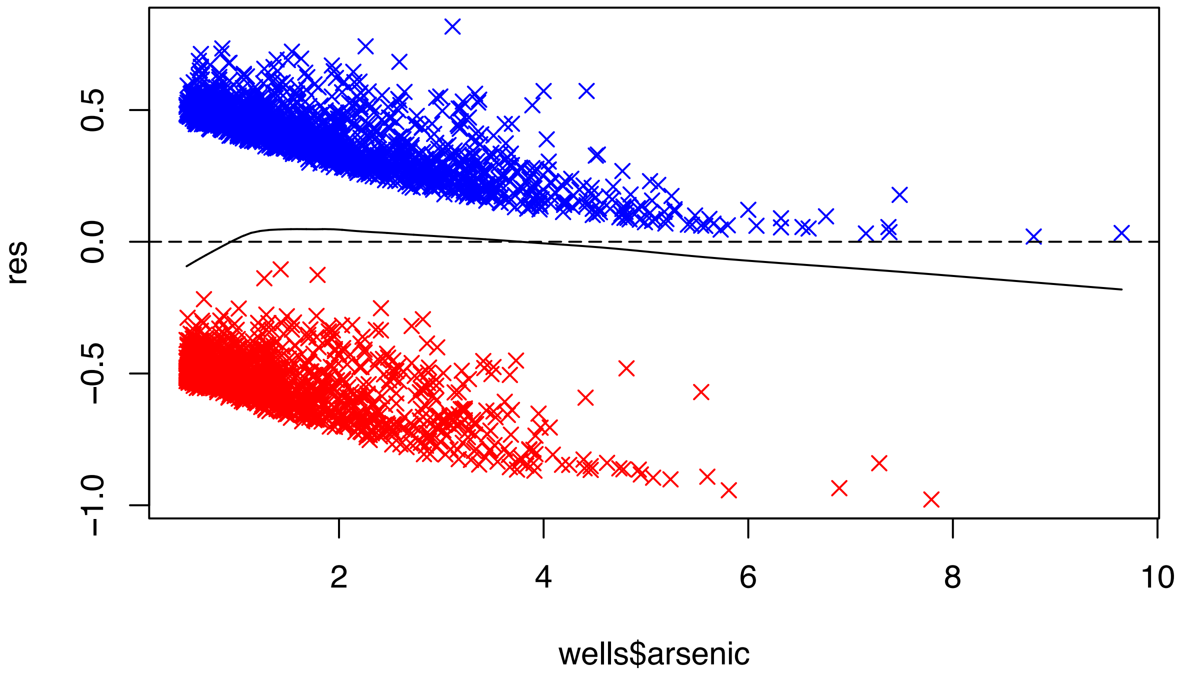
```
# First a plot of residuals agains distance
plot(res ~ wells$dist, col=ifelse(res>0, "blue","red"), pch=4)
lines(lowess(res ~ wells$dist))
abline(h=0, lty=5, col="black")
```



Plotting the residuals against well distance, we see the two groups of residuals, that are given since y_i can only be zero or one, whereas $\hat{\tilde{y_i}}$ is an average, and as such, is between zero or one, so the residuals are grouped depending on whether y_i is zero or one. Nonetheless, by plotting with `lowess` and thus, weighting the residuals, we see the line is very close to zero, meaning the residuals do not vary much with distance.

Now, to plot the residuals against arsenic concentration:

```
plot(res ~ wells$arsenic, col=ifelse(res>0, "blue","red"), pch=4)
lines(lowess(res ~ wells$arsenic))
abline(h=0, lty=5, col="black")
```



The line is close to zero, although it does tend downwards as the concentration of arsenic increases. Nonetheless, since the data is sparse at far distances, this may simply be driven by a handful of outliers.

Back to the quesion

Then, because this is logistic regression with binary outcomes, consider how to best display the residuals. Note that just plotting residuals will not result in an informative plot because the y's are binary.

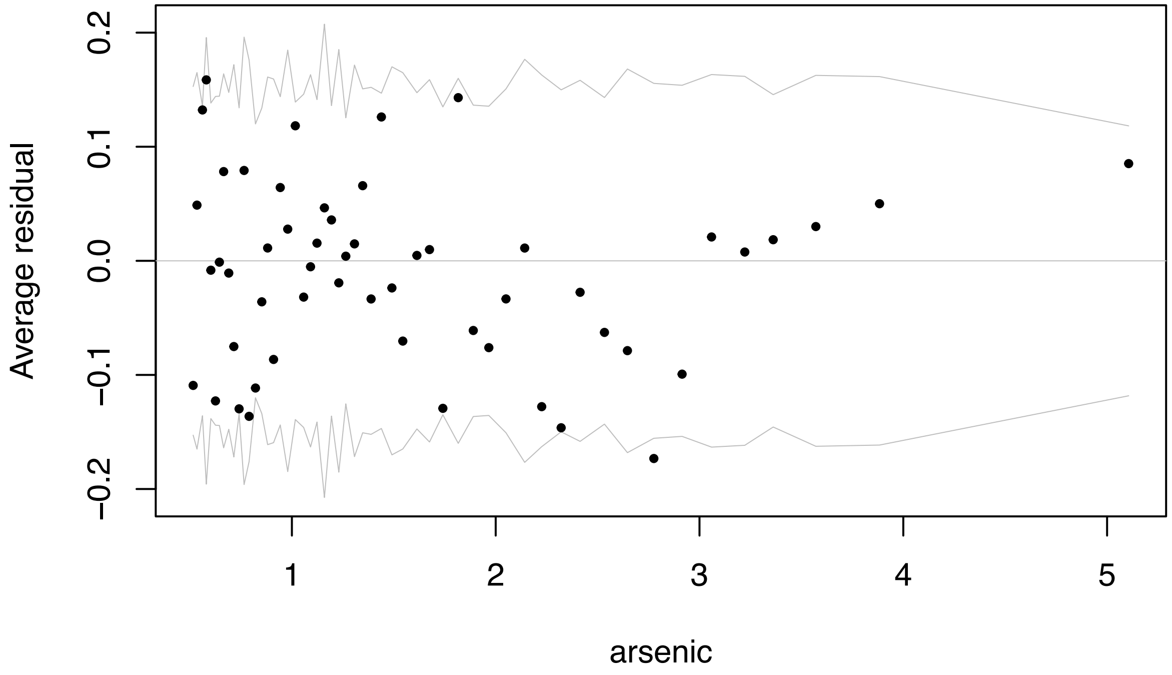
Alvaro comment: That said, by using `lowess`, we get a better sense of how they are distributed.

You may be able to find better resources but in case it's still helpful, in my pre-tidyverse and ggplot life, I have used a function from GH for plotting residuals

```
#----
# function for binned residual plots from GH
#-----
binned.resids <- function (x, # what to bin over?
                           y, # what to bin, eg. residuals
                           nclass=sqrt(length(x))){
  breaks.index <- floor(length(x)*(1:(nclass-1))/nclass)
  breaks <- c (-Inf, sort(x)[breaks.index], Inf)
  output <- NULL
  xbreaks <- NULL
  x.binned <- as.numeric (cut (x, breaks))
  for (i in 1:nclass){
    items <- (1:length(x))[x.binned==i]
    x.range <- range(x[items])
    xbar <- mean(x[items])
    ybar <- mean(y[items])
    n <- length(items)
    sdev <- sd(y[items])
    output <- rbind (output, c(xbar, ybar, n, x.range, 2*sdev/sqrt(n)))
  }
  colnames (output) <- c ("xbar", "ybar", "n", "x.lo", "x.hi", "2se")
  return (list (binned=output, xbreaks=xbreaks))
}
```

Example use for made up residuals

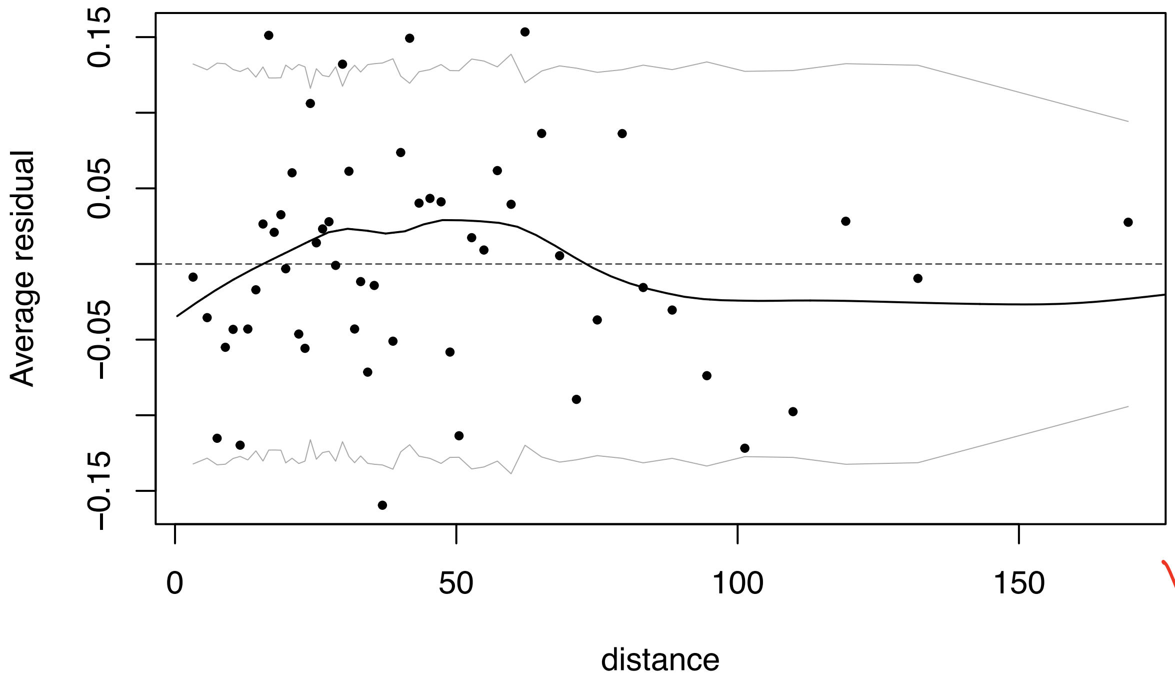
```
n <- length(wells$y)
resid <-  runif(n, -1,1)# just making up something
result <- data.frame(binned.resids(wells$arsenic, resid)$binned)
plot(range(result$xbar), range(result$ybar,result$X2se, -result$X2se),
     ylab="Average residual", type="n", xlab = "arsenic")
abline (0,0, col="gray", lwd=.5)
lines (result$xbar, result$X2se, col="gray", lwd=.5)
lines (result$xbar, -result$X2se, col="gray", lwd=.5)
points (result$xbar,result$ybar, pch=19, cex=.5)
```



✓

Answer
Now, let me try, first for distance:

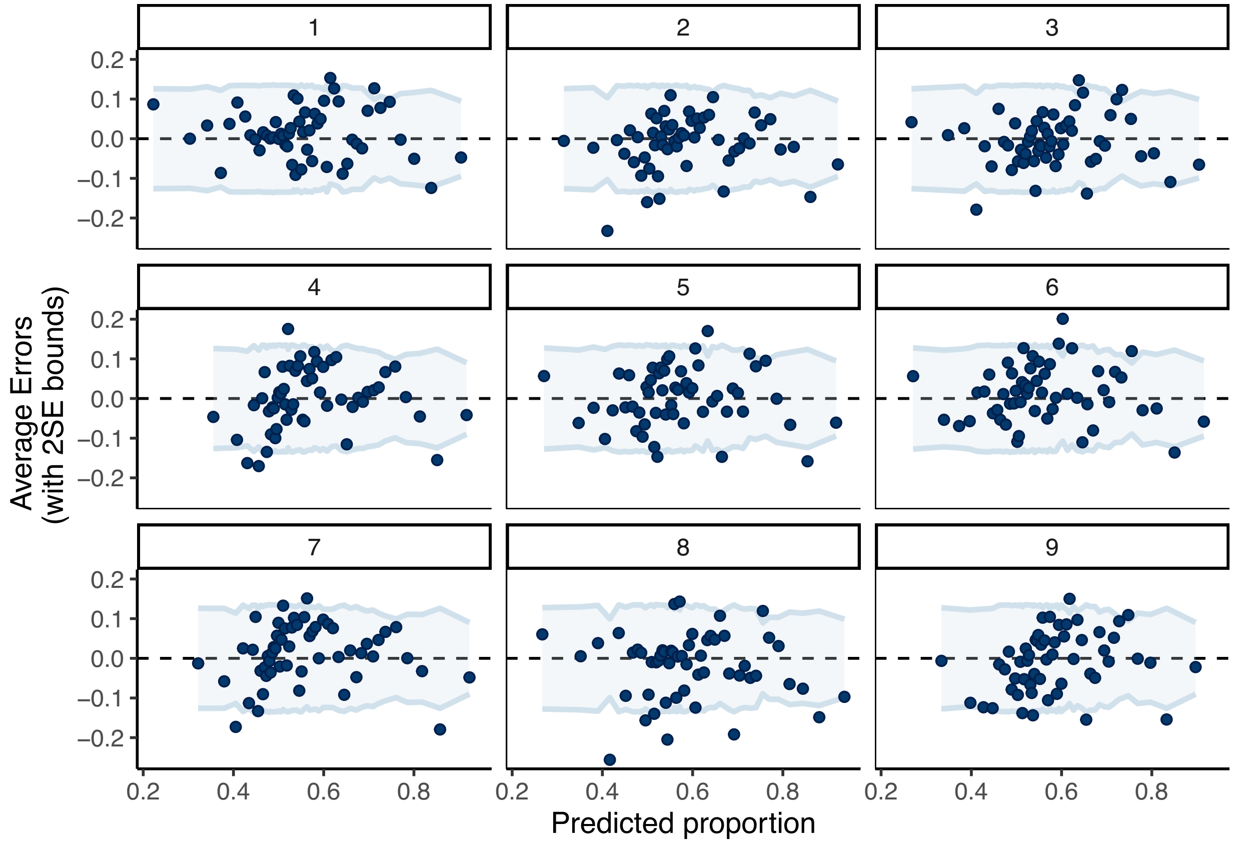
```
n <- length(wells$y)
resid <- res
result <- data.frame(binned.resids(wells$dist, res)$binned)
plot(range(result$xbar), range(result$ybar,result$X2se, -result$X2se),
     ylab="Average residual", type="n", xlab = "distance")
abline (0,0, col="black", lwd=.5, lty=5)
lines (result$xbar, result$X2se, col="darkgray", lwd=.5)
lines (result$xbar, -result$X2se, col="darkgray", lwd=.5)
points (result$xbar,result$ybar, pch=19, cex=.5)
lines(lowess(res ~ wells$dist))
```



✓

There are five points (out of 54) outside the error bars, which means 90% are within our error margins. There is no obvious pattern, and the results seem reasonable. Now, a different kind of plot:

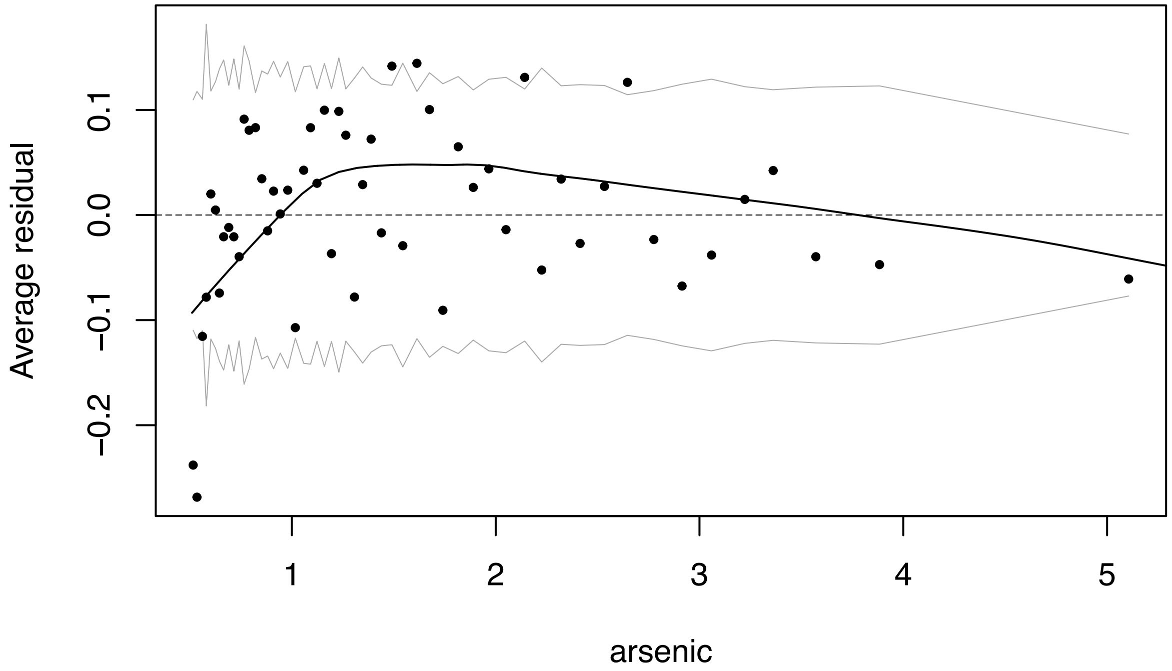
```
pp_check(hw4q2_3.fit, type = "error_binned", ndraws = 9, x="c_dist100") + theme_classic()
```



Similarly to before, most observations seem to be within the error bars, with no obvious pattern.

Now, for the arsenic plots:

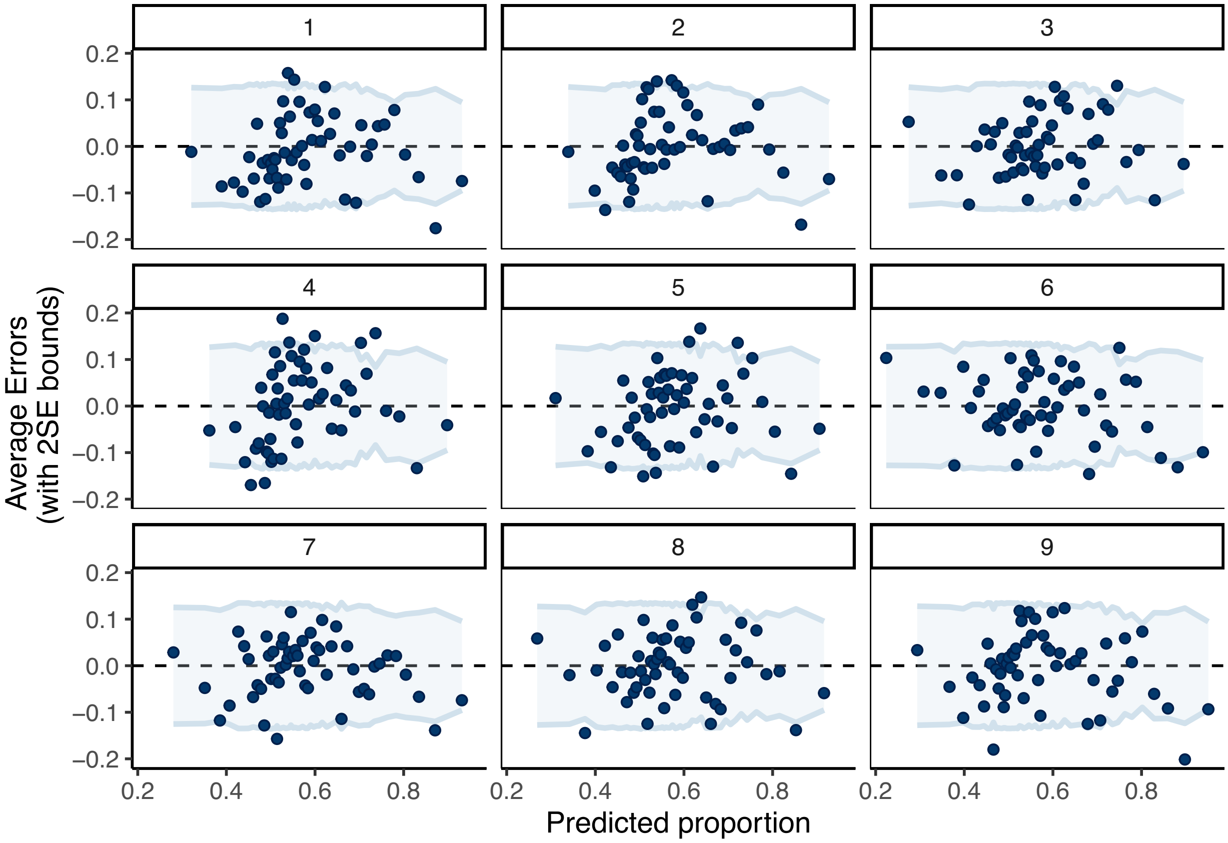
```
n <- length(wells$y)
resid <- res
result <- data.frame(binned.resids(wells$arsenic, res)$binned)
plot(range(result$xbar), range(result$ybar,result$X2se, -result$X2se),
     ylab="Average residual", type="n", xlab = "arsenic")
abline (0,0, col="black", lwd=.5, lty=5)
lines (result$xbar, result$X2se, col="darkgray", lwd=.5)
lines (result$xbar, -result$X2se, col="darkgray", lwd=.5)
points (result$xbar,result$ybar, pch=19, cex=.5)
lines(lowess(res ~ wells$arsenic))
```



There are six points (out of 54) outside the error bars, which means roughly 90% are within our error margins. As was mentioned before, there seems to be an increase and then gradual decrease in the residuals (as viewed with `lowess`), but the line seems close enough to zero to be reasonable. Even so, at the lowest values of arsenic, it does seem the model is overpredicting the probability of a switch (given by the very negative residuals), since the lowest points are far below the error bars.

Now, a different kind of plot:

```
pp_check(hw4q2_3.fit, type = "error_binned", ndraws = 9, x="c_arsenic") + theme_classic()
```



This looks almost identical to the previous `pp_check` plot, and I'm not convinced it is plotting by variables correctly, so I will refrain from any interpretation.

Question 4: Posterior predictive check

10

The fit of model (3) is not great for low values of arsenic: the probability of switching is overpredicted at very low arsenic levels. To improve model diagnostics, let's consider another model (model 4) where arsenic levels are log-transformed:

$$\begin{aligned} y_i &\sim Bern(p_i), \\ logit(p_i) &= \beta_0 + \beta_1 \cdot (d_i - \bar{d}) + \beta_2 \cdot (a_i^* - \bar{a}_i^*) + \beta_3 \cdot (d_i - \bar{d})(a_i^* - \bar{a}^*), \text{ for model 4} \end{aligned}$$

where a_i^* refers to log-transformed arsenic.

Suppose that one of the outcomes of interest in this study is predicting whether or not a household that is using a well with "unsafe but relatively low arsenic levels'' (say arsenic levels up to 0.82, which is the 25th percentile of the observed sample of arsenic values) will switch. Carry out a posterior predictive check to verify whether model (3) with arsenic and/or model (4) with log(arsenic) give a reasonable prediction for the proportion of switching households (with arsenic levels less than 0.82).

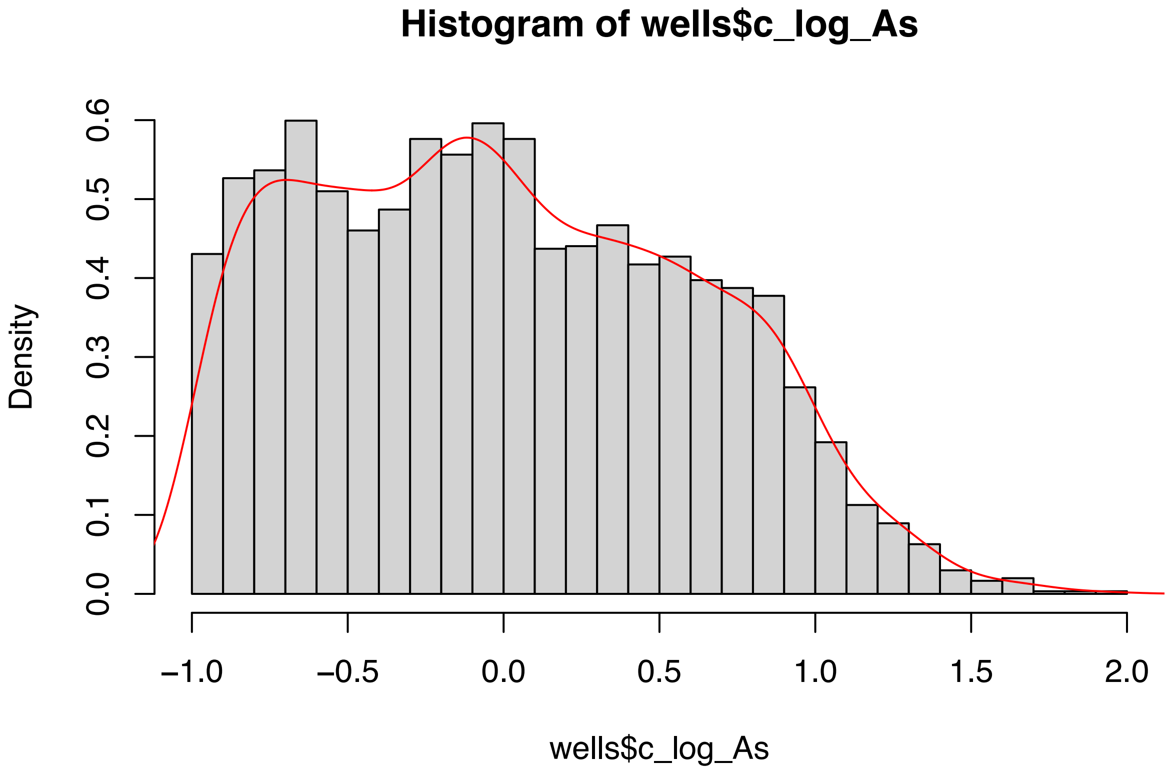
Hint: specify a summary statistic $T(\boldsymbol{y})$ that summarizes the outcome of interest and calculate $T(\boldsymbol{y})$ for the data set. Then construct replicated data sets $\tilde{\boldsymbol{y}}^{(s)}$ with summary statistics $T(\tilde{\boldsymbol{y}}^{(s)})$ and evaluate how extreme $T(\boldsymbol{y})$ is compared to the sample of $T(\tilde{\boldsymbol{y}}^{(s)})$'s.

Answer
First, I will create a centered, log-transformed arsenic variable:

```
wells <- wells %>%
  mutate(log_As = log(arsenic),
         c_log_As = log_As - mean(log_As))
summary(wells[,10:11])
```

```
##      log_As            c_log_As      
##  Min.   :-0.6733   Min.   :-0.9872  
##  1st Qu.:-0.1985   1st Qu.:-0.5123  
##  Median : 0.2624   Median :-0.0515  
##  Mean   : 0.3139   Mean   : 0.0000  
##  3rd Qu.: 0.7885   3rd Qu.: 0.4746  
##  Max.   : 2.2670   Max.   : 1.9531  
```

```
hist(wells$c_log_As, breaks = 30, freq = FALSE)
lines(density(wells$c_log_As), col = "red")
```



```
# Not exactly a uniform distribution - there are a few extreme outliers with very high values for log(A
```

Now, to fit the model:

```
hw4q4.fit <- brm(formula = y ~ 1 + c_dist100 * c_log_As,
                 file = "output/hw4q4",
                 data = wells, sample_prior = T,
                 prior = c(t_prior),
                 family = bernoulli(link = "logit"),
                 chains = 4, iter = 2000, warmup = 1000,
                 cores = getOption("mc.cores", 4),
                 thin = 1, seed = 1234)
```

Now the results:

```
# Reviewing output and diagnostics
summary(hw4q4.fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ 1 + c_dist100 * c_log_As
##    Data: wells (Number of observations: 3020)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Population-Level Effects:
##                    Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept              0.34      0.04     0.27     0.42 1.00     3915     3126
## c_dist100             -0.95      0.11    -1.17    -0.74 1.00     3731     3275
## c_log_As               0.88      0.07     0.74     1.01 1.00     3632     2909
## c_dist100:c_log_As    -0.23      0.18    -0.60     0.12 1.00     4030     2974
##
## 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).
```

```
# Fixed effects
summary(hw4q4.fit)$fixed
```

```
##                      Estimate  Est.Error   l-95% CI   u-95% CI     Rhat
## Intercept           0.3431843 0.04051240  0.2651958  0.4216767 1.001260
## c_dist100          -0.9514470 0.10927986 -1.1728785 -0.7436392 1.001325
## c_log_As            0.8755819 0.06682082  0.7433851  1.0100147 1.000857
## c_dist100:c_log_As -0.2321066 0.18239534 -0.6034037  0.1194464 1.001988
##                    Bulk_ESS Tail_ESS
## Intercept          3915.481 3126.279
## c_dist100          3730.804 3274.918
## c_log_As           3632.312 2909.386
## c_dist100:c_log_As 4029.859 2974.208
```

```
# Intercept and coefficients
fixef(hw4q4.fit)
```

```
##                      Estimate  Est.Error       Q2.5      Q97.5
## Intercept           0.3431843 0.04051240  0.2651958  0.4216767
## c_dist100          -0.9514470 0.10927986 -1.1728785 -0.7436392
## c_log_As            0.8755819 0.06682082  0.7433851  1.0100147
## c_dist100:c_log_As -0.2321066 0.18239534 -0.6034037  0.1194464
```

```
exp(fixef(hw4q4.fit)[,-2])
```

```
##                     Estimate      Q2.5     Q97.5
## Intercept          1.4094285 1.3036863 1.5245156
## c_dist100          0.3861818 0.3094748 0.4753807
## c_log_As           2.4002717 2.1030426 2.7456413
## c_dist100:c_log_As 0.7928616 0.5469468 1.1268729
```

It's helpful to look at the residuals:

```
ynew_si <- posterior_predict(hw4q4.fit)
dim(ynew_si)
```

```
## [1] 4000 3020
```

```
ytildehat_i <- apply(ynew_si, 2, mean)
length(ytildehat_i)
```

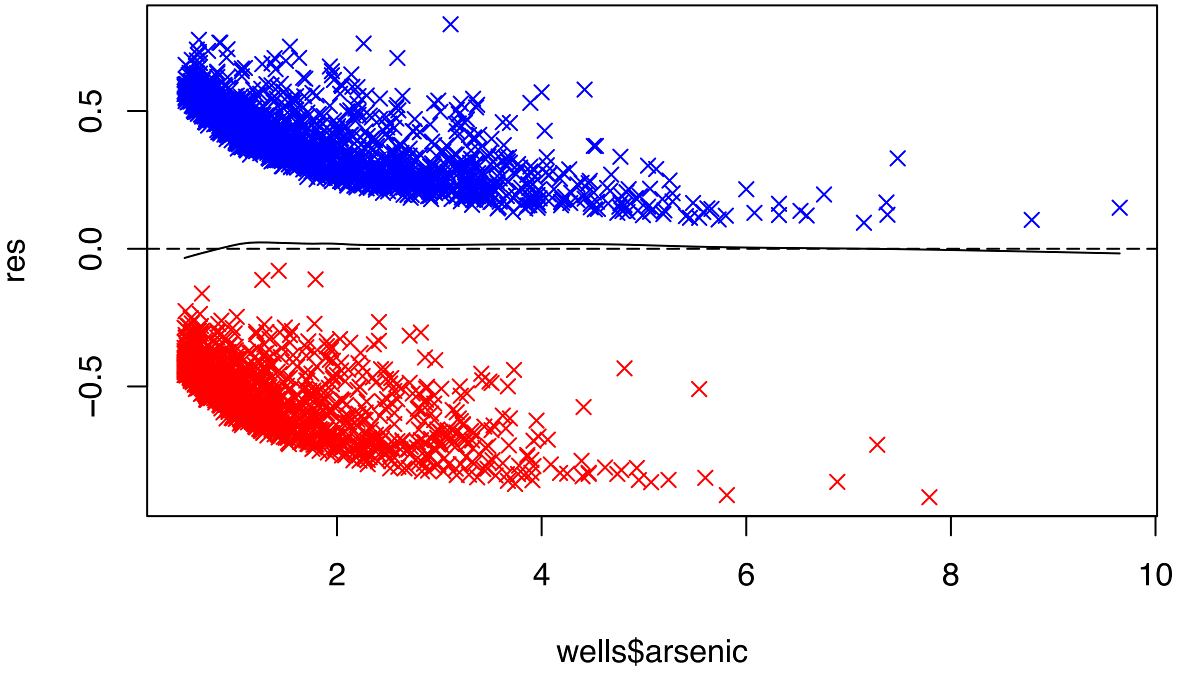
```
## [1] 3020
```

```
res <- wells$y - ytildehat_i
summary(res)
```

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.9022500 -0.4926250  0.2427500  0.0000339  0.4226250  0.8150000
```

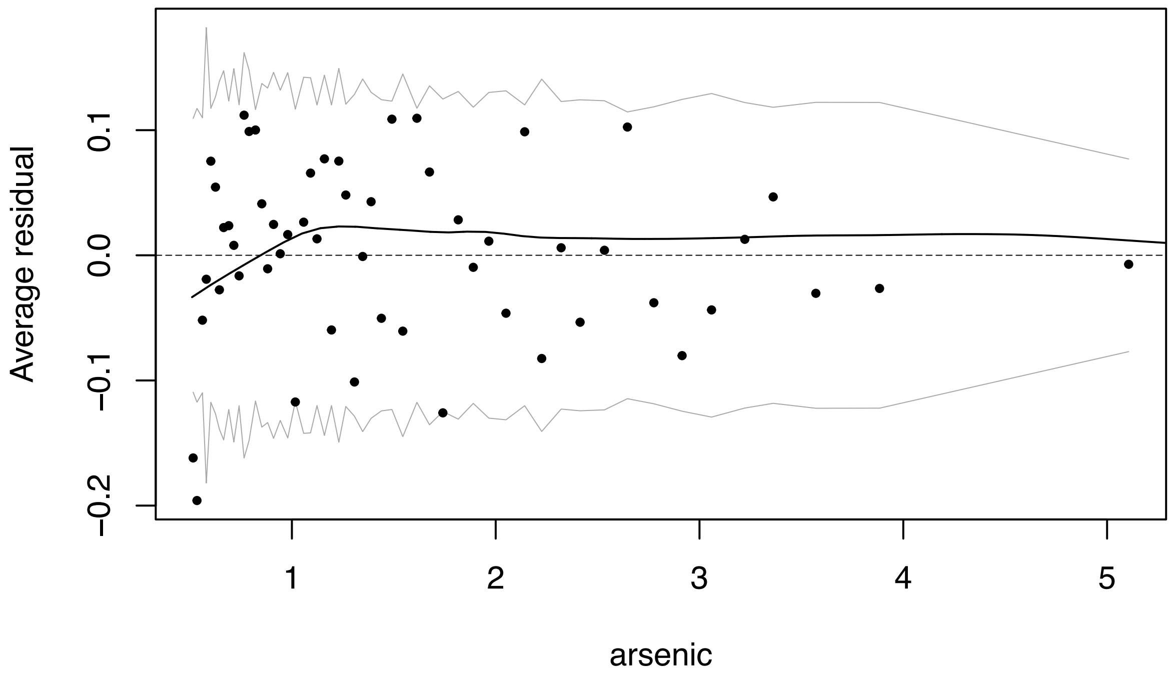
First residual plot:

```
# First a plot of residuals agains distance
plot(res ~ wells$arsenic, col=ifelse(res>0, "blue","red"), pch=4)
lines(lowess(res ~ wells$arsenic))
abline(h=0, lty=5, col="black")
```



The line fit does look better than before, now for a plot of the binned residuals:

```
n <- length(wells$y)
resid <- res
result <- data.frame(binned.resids(wells$arsenic, res)$binned)
plot(range(result$xbar), range(result$ybar,result$X2se, -result$X2se),
     ylab="Average residual", type="n", xlab = "arsenic")
abline (0,0, col="black", lwd=.5, lty=5)
lines (result$xbar, result$X2se, col="darkgray", lwd=.5)
lines (result$xbar, -result$X2se, col="darkgray", lwd=.5)
points (result$xbar,result$ybar, pch=19, cex=.5)
lines(lowess(res ~ wells$arsenic))
```



Most values now lie within the error bars, although there still are two extreme values at low arsenic levels. So, we seem to have improved the prediction somewhat, although the shape and pattern is similar to the normal arsenic values.

Now, on to the posterior predictive check. First, as per the suggestion, I will use a summary statistic $T(\mathbf{y})$ to summarize the outcome of interest. Namely:

$$T(\mathbf{y}) = (\bar{\theta}|\mathbf{arsenic} < \mathbf{0.82})$$

✓

In my dataset this is given by:

```
small_As <- wells %>% filter(arsenic < 0.82)
(T_y <- mean(small_As$y))
```

```
## [1] 0.4338336
```

```
# 0.4338336
```

Now, replicated datasets for comparison and estimated $T(\mathbf{y})$s:

```r
set.seed(1234)

# Model (3)
ynew_si_3 <- posterior_predict(hw4q2_3.fit, newdata = small_As)
dim(ynew_si_3)
```

```
## [1] 4000  733
```

```r
ytildehat_i_3 <- apply(ynew_si_3, 2, mean)
length(ytildehat_i_3)
```

```
## [1] 733
```

```r
(T_y_3 <- mean(ytildehat_i_3))
```

```
## [1] 0.491294
```

```r
# 0.4918704
# What percentile would the real mean fall in a distribution of our sampled means?
(ecdf(ytildehat_i_3)(T_y))
```

```
## [1] 0.1091405
```

```r
# 0.111869 - the 11th percentile, which is getting to the tail of our distribution

# Model (4)
ynew_si_4 <- posterior_predict(hw4q4.fit, newdata = small_As)
dim(ynew_si_4)
```

```
## [1] 4000  733
```

```r
ytildehat_i_4 <- apply(ynew_si_4, 2, mean)
length(ytildehat_i_4)
```

```
## [1] 733
```

```r
(T_y_4 <- mean(ytildehat_i_4))
```

```
## [1] 0.4459887
```

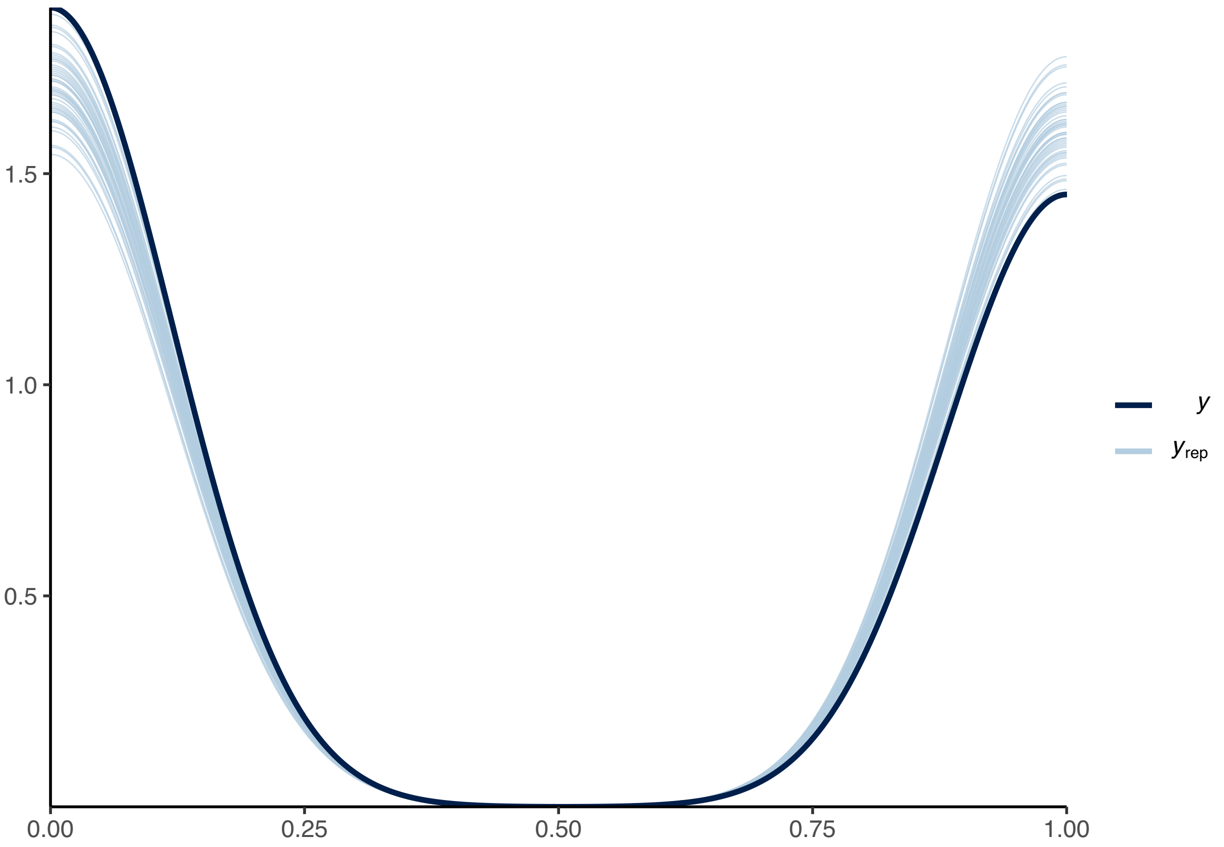
```r
# 0.4459986
# What percentile does the real mean fall in Model (4)
(ecdf(ytildehat_i_4)(T_y))
```

```
## [1] 0.3547067
```

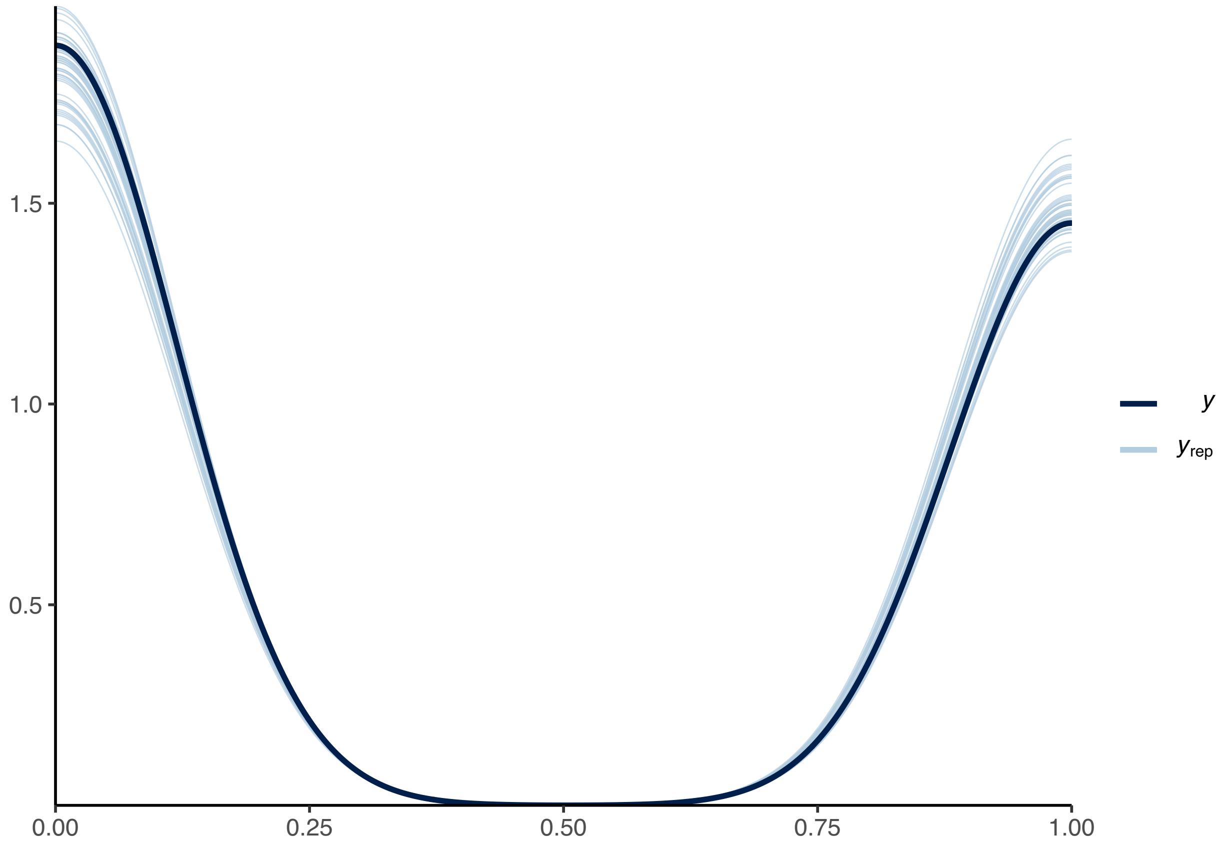
```
# 0.3560709 - the 36th percentile, which is much better
```

Now, for some plots:

```
set.seed(1234)
# Model (3)
select_samp_3 <- sample(1:dim(ynew_si_3)[1], 50)
ppc_dens_overlay(y = small_As$y, yrep = ynew_si_3[select_samp_3, ]) + theme_classic()
```



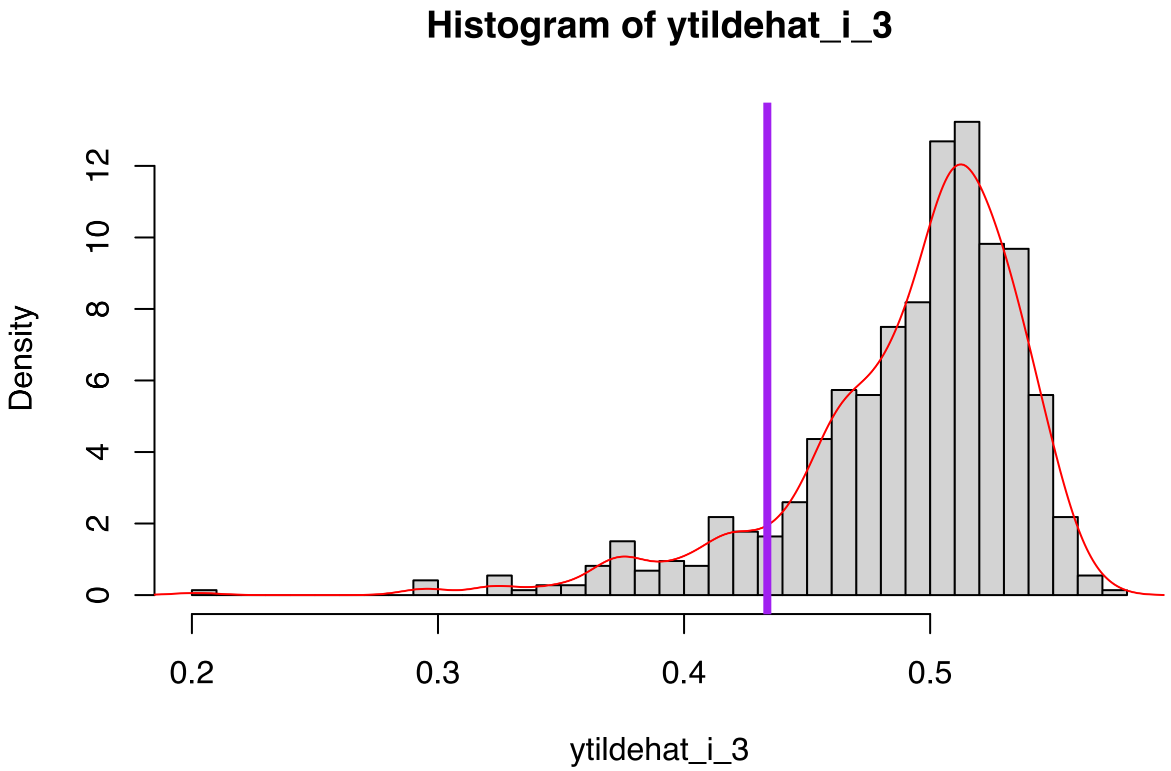
```
# Model (4)
select_samp_4 <- sample(1:dim(ynew_si_4)[1], 50)
ppc_dens_overlay(y = small_As$y, yrep = ynew_si_4[select_samp_4, ]) + theme_classic()
```



For Model (3), the repetitions don't quite match the truth - underpredicting those who don't switch and overpredicting those who do switch. The difference is notable and dramatic. Model (4) shows a more even distribution on both sides of the real values, implying that Model (4) is a better fit for low values of arsenic.

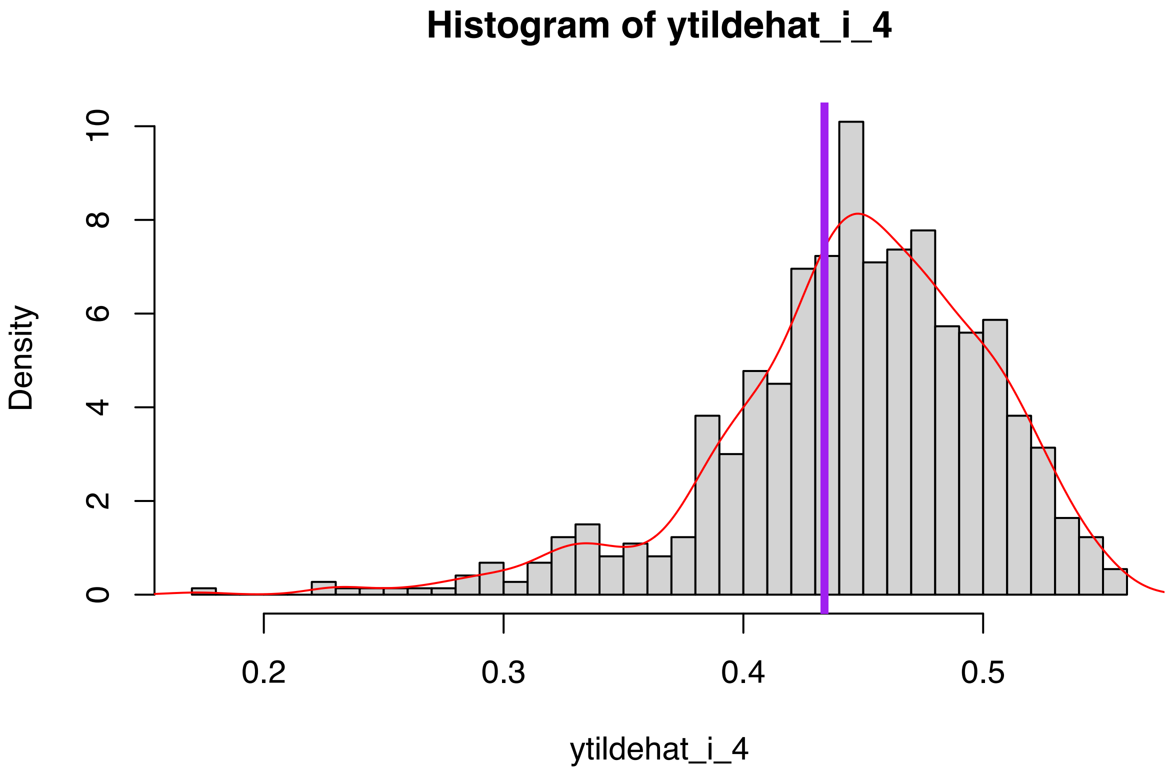
Now, to check the distribution of sample means in both models compared to the mean in the `wells` dataset:

```
# Model (3)
hist(ytildehat_i_3, breaks = 30, freq = FALSE)
lines(density(ytildehat_i_3), col = "red")
abline(v = T_y, col = "purple", lwd = 4)
```



As seen previously, this line lies towards the tail of our distribution of means.

```
# Model (4)
hist(ytildehat_i_4, breaks = 30, freq = FALSE)
lines(density(ytildehat_i_4), col = "red")
abline(v = T_y, col = "purple", lwd = 4)
```



Here the real mean lies much closer to the center of the distribution of our sampled means.

Conclusion: Model (4) gives a *much* better prediction for the proportion of switching households than Model (3). ✓

Question 5: Multilevel logistic regression (extra credit) 10

According to GH 14.6 (Q2), the observations are obtained in different villages, which makes for a nice extension of the logistic regression model into a multilevel logistic regression model. However, I was not able to find the village grouping in the data sets provided online. To not deprive you from this nice extension and let you fit a multilevel logistic model, go ahead and construct your own groupings as follows:

```
set.seed(12345)
n <- length(wells$y)
# assign households to villages
J <- 300
getj1_i <- c(seq(1,J), sample(size = n-J, x = seq(1,J), replace = TRUE))
getj2_i <- sort(getj1_i) # now the households are assumed to be sorted by village
```

where the first grouping (summarized in 'getj1.i') is random while in the second grouping, the households are grouped in the order at which they appear in the dataset.

Write out in equations an extension for model (2), where each group has its own intercept, that is estimated hierarchically. Then fit the model, using both groupings (so fit the same model twice).

Comment on the difference in resulting fits between using grouping 1 and grouping 2. In particular, do you have any thoughts on why the across-village variance in intercept is smaller for the 1st grouping as compared to the second grouping?

Answer
The question asks us to write equations for models with different intercepts, but not slopes, and so I will limit myself to differences in the intercept.

$$y_i|\theta_i \sim Bern(\theta_i),$$
$$logit(\theta_i) = \alpha_{j[i]} + \beta_1 \cdot (d_i - \bar{d}) + \beta_2 \cdot (a_i - \bar{a}),$$
$$\alpha_j|\mu_\alpha, \sigma_\alpha \sim N(\mu_\alpha, \sigma_\alpha)$$

✓

Now, to fit the the model for grouping 1, Model g1, first I should organize the data:

```
wells_g1 <- wells %>% add_column(village = getj1_i)
```

Now, to fit the model:

```
hw4q5_1.fit <- brm(formula = y ~ (1 | village) + c_dist100 + c_arsenic,
                   file = "output/hw4q5_1",
                   data = wells_g1, sample_prior = T,
                   prior = c(t_prior),
                   family = bernoulli(link = "logit"),
                   chains = 4, iter = 2000, warmup = 1000,
                   cores = getOption("mc.cores", 4),
                   thin = 1, seed = 1234)
```

✓

Now the results:

```
# Reviewing output and diagnostics
summary(hw4q5_1.fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ (1 | village) + c_dist100 + c_arsenic
##    Data: wells_g1 (Number of observations: 3020)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Group-Level Effects:
## ~village (Number of levels: 300)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     0.18      0.09     0.01     0.36 1.01      794     1182
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.34      0.04     0.26     0.42 1.00     5824     2829
## c_dist100    -0.90      0.11    -1.11    -0.70 1.00     6893     2847
## c_arsenic     0.47      0.04     0.39     0.55 1.00     6586     2932
##
## 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).
```

```
# Fixed effects
summary(hw4q5_1.fit)$fixed
```

```
##              Estimate  Est.Error   l-95% CI   u-95% CI      Rhat Bulk_ESS
## Intercept  0.3371508 0.04082730  0.2587883  0.4175954 1.0004736 5824.451
## c_dist100 -0.9045178 0.10597001 -1.1104176 -0.7003928 0.9999831 6893.220
## c_arsenic  0.4657696 0.04075831  0.3862826  0.5456036 1.0012485 6586.082
##           Tail_ESS
## Intercept 2828.568
## c_dist100 2846.679
## c_arsenic 2932.052
```

```
# Intercept and coefficients
fixef(hw4q5_1.fit)
```

```
##              Estimate  Est.Error       Q2.5      Q97.5
## Intercept  0.3371508 0.04082730  0.2587883  0.4175954
## c_dist100 -0.9045178 0.10597001 -1.1104176 -0.7003928
## c_arsenic  0.4657696 0.04075831  0.3862826  0.5456036
```

```
exp(fixef(hw4q5_1.fit)[,-2])
```

```
##           Estimate      Q2.5     Q97.5
## Intercept 1.400950 1.2953596 1.5183063
## c_dist100 0.404737 0.3294214 0.4963903
## c_arsenic 1.593240 1.4715005 1.7256497
```

The results are very similar to what we obtained with Model (2), although the variance is slightly larger in Model g1.

Now for the second grouping, Model g2:

```
wells_g2 <- wells %>% add_column(village = getj2_i)
```

Now, to fit the model:

```
hw4q5_2.fit <- brm(formula = y ~ (1 | village) + c_dist100 + c_arsenic,
                   file = "output/hw4q5_2",
                   data = wells_g2, sample_prior = T,
                   prior = c(t_prior),
                   family = bernoulli(link = "logit"),
                   chains = 4, iter = 2000, warmup = 1000,
                   cores = getOption("mc.cores", 4),
                   thin = 1, seed = 1234)
```

✓

Now the results:

```
# Reviewing output and diagnostics
summary(hw4q5_2.fit)
```

```
##  Family: bernoulli
##   Links: mu = logit
## Formula: y ~ (1 | village) + c_dist100 + c_arsenic
##    Data: wells_g2 (Number of observations: 3020)
##   Draws: 4 chains, each with iter = 2000; warmup = 1000; thin = 1;
##          total post-warmup draws = 4000
##
## Group-Level Effects:
## ~village (Number of levels: 300)
##               Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## sd(Intercept)     1.03      0.08     0.89     1.19 1.00     1577     2517
##
## Population-Level Effects:
##           Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
## Intercept     0.42      0.08     0.27     0.57 1.00     2229     2848
## c_dist100    -0.90      0.13    -1.17    -0.63 1.00     4733     3301
## c_arsenic     0.44      0.05     0.34     0.54 1.00     5345     3214
##
## 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).
```

```
# Fixed effects
summary(hw4q5_2.fit)$fixed
```

```
##             Estimate  Est.Error   l-95% CI   u-95% CI      Rhat Bulk_ESS
## Intercept  0.4223997 0.07544968  0.2727168  0.5666961 1.0012239 2228.679
## c_dist100 -0.9027109 0.13494487 -1.1721133 -0.6321435 0.9997463 4733.139
## c_arsenic  0.4378278 0.05000637  0.3408190  0.5356309 0.9997885 5345.441
##           Tail_ESS
## Intercept 2848.194
## c_dist100 3301.225
## c_arsenic 3213.588
```

```
# Intercept and coefficients
fixef(hw4q5_2.fit)
```

```
##             Estimate  Est.Error       Q2.5      Q97.5
## Intercept  0.4223997 0.07544968  0.2727168  0.5666961
## c_dist100 -0.9027109 0.13494487 -1.1721133 -0.6321435
## c_arsenic  0.4378278 0.05000637  0.3408190  0.5356309
```

```
exp(fixef(hw4q5_2.fit)[,-2])
```

```
##           Estimate      Q2.5     Q97.5
## Intercept 1.525618 1.3135282 1.7624345
## c_dist100 0.405469 0.3097117 0.5314514
## c_arsenic 1.549338 1.4060987 1.7085258
```

The results here are relatively similar to those of Model (2) and Model g1, but the most important difference is at the level of the intercept μ_α. It is much bigger than the other two models (0.42 vs 0.33, 0.34), and the standard error is twice as big (0.08 vs. 0.04). The reason is that the data are probably not organized in random fashion, and so, by grouping "villages" in the order they appear on the dataset, they are being non-randomly assigned to different villages. This means that the group means will also vary more than those that are completely random (because random allocation will lead to less variance), and this will be picked up by the hierarchical model. That is also why the between-group (village) standard deviation is *much* larger in model g2 (almost six times as large: 1.03 vs 0.18), since the group means are different in a meaningful way because of the non-random allocation. ✓

In summary, neighboring rows are more similar to each other than random rows, which means that group means will cluster in patterns that differentiate them, and make them more different between each other. This means more between group variance, but also means the standard error for the intercept is larger.