HW 1 Solutions

Chen Xu BUID:49903384

9/15/2021

7.2 Fake-data simulation and regression:

Simulate 100 data points from the linear model, y = a + bx + error, with a = 5, b = 7, the values of x being sampled at random from a uniform distribution on the range [0, 50], and errors that are normally distributed with mean 0 and standard deviation 3.

7.2a

Fit a regression line to these data and display the output.

```
x_72a <- runif(100,0,50)
error_72a <- rnorm(100, mean = 0,sd = 3)
a <- 5
b <- 7
y_72a <- a + b*x_72a + error_72a
fitmod_72a <- lm(y_72a ~ x_72a)
summary(fitmod_72a)
```

```
##
## Call:
## lm(formula = y_72a ~ x_72a)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1657 -2.0972  0.1255  1.9172  9.2662
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.50842    0.57264   7.873 4.73e-12 ***
## x_72a        7.00276    0.02001 349.990  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.015 on 98 degrees of freedom
## Multiple R-squared:  0.9992, Adjusted R-squared:  0.9992
## F-statistic: 1.225e+05 on 1 and 98 DF,  p-value: < 2.2e-16
```

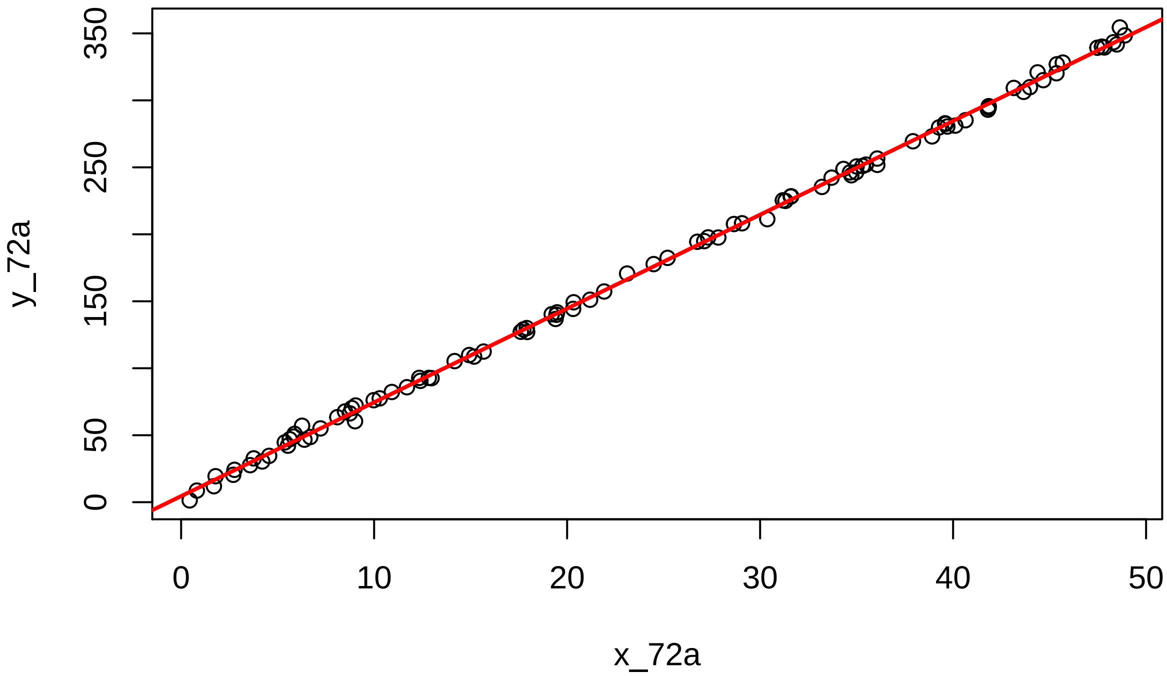
```
fitmod_72a
```

```
##
## Call:
## lm(formula = y_72a ~ x_72a)
##
## Coefficients:
## (Intercept)        x_72a
##       4.508        7.003
```

7.2b

Graph a scatterplot of the data and the regression line.

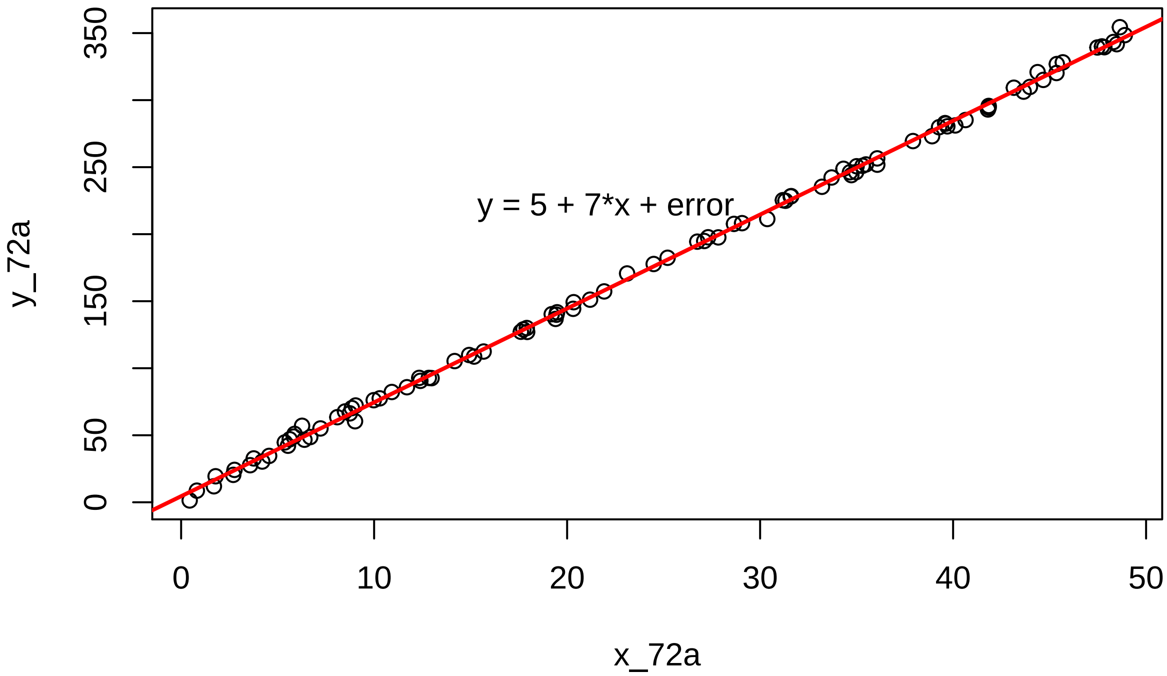
```
plot(x_72a,y_72a)
abline(fitmod_72a, col = "red", lwd = 2)
```



7.2c

Use the text function in R to add the formula of the fitted line to the graph.

```
plot(x_72a,y_72a)
abline(fitmod_72a, col = "red", lwd = 2)
text(22,220, "y = 5 + 7*x + error")
```



7.3 Fake-data simulation and fitting the wrong model:

Simulate 100 data points from the model, y = a + bx + cx2 + error, with the values of x being sampled at random from a uniform distribution on the range [0, 50], errors that are normally distributed with mean 0 and standard deviation 3, and a, b, c chosen so that a scatterplot of the data shows a clear nonlinear curve.

7.3 a

Fit a regression line stan_glm(y ~ x) to these data and display th'e output.

```
x_73a <- runif(100,0,50)
error_73a <- rnorm(100, mean = 0,sd = 3)
a = 3
b = 2
c = 1
y_73a <- a + b*x_73a + c*x_73a^2 +error_73a
fit_73a <- stan_glm(y_73a ~ x_73a)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 8.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.83 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
```

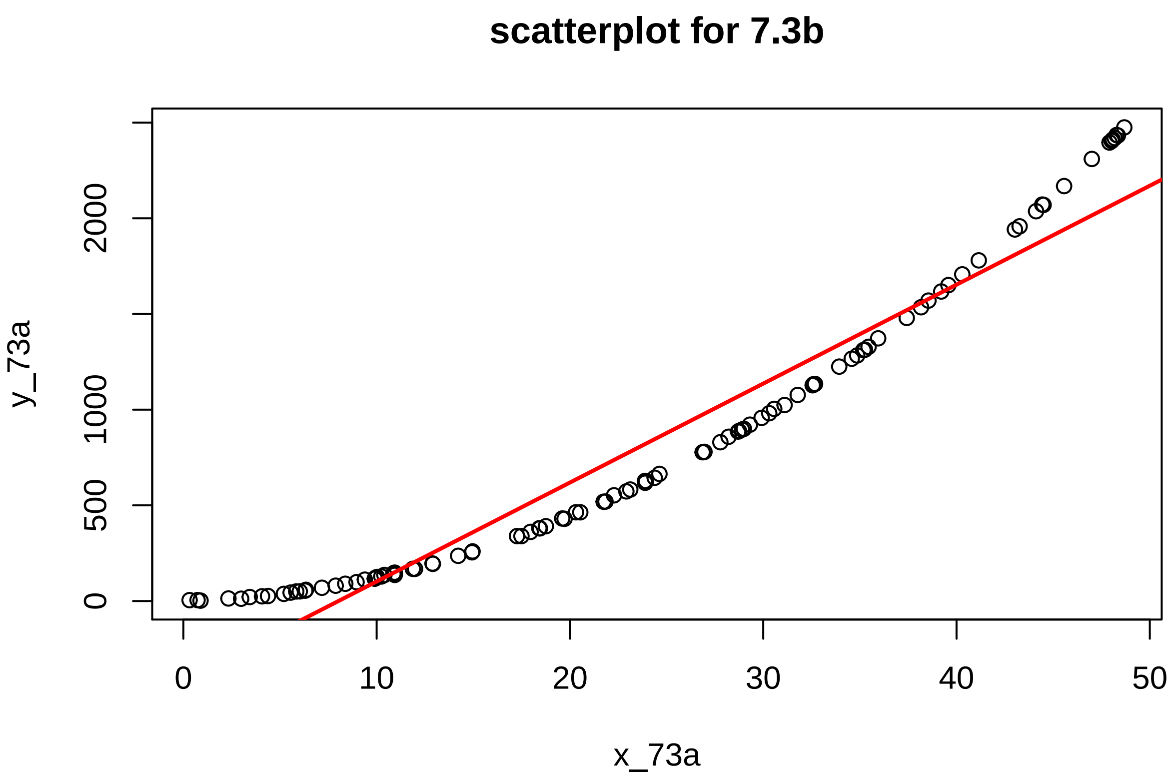
```
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.248506 seconds (Warm-up)
## Chain 1:                0.053337 seconds (Sampling)
## Chain 1:                0.301843 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.195627 seconds (Warm-up)
## Chain 2:                0.068048 seconds (Sampling)
## Chain 2:                0.263675 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.1e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
```

```
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.524722 seconds (Warm-up)
## Chain 3:                0.058019 seconds (Sampling)
## Chain 3:                0.582741 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.1e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.11 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.234751 seconds (Warm-up)
## Chain 4:                0.058676 seconds (Sampling)
## Chain 4:                0.293427 seconds (Total)
## Chain 4:
```

7.3b

Graph a scatterplot of the data and the regression line. This is the best-fit linear regression. What does "best-fit" mean in this context?

```
data73 <- data.frame(y_73a,x_73a)
plot(main = "scatterplot for 7.3b",x_73a, y_73a)
abline(fit_73a, col = "red", lwd = 2)
```



The "best-fit" mean in this context is that this regression line for this plot has the Least sum of squares error.

7.6 Formulating comparisons as regression models:

Take the election forecasting model and simplify it by creating a binary predictor defined as x = 0 if income growth is less than 2% and x = 1 if income growth is more than 2%.

```
ghv_data_dir <- "https://raw.githubusercontent.com/avehtari/ROS-Examples/master/"
efm <- read.table (paste0(ghv_data_dir,"ElectionsEconomy/data/hibbs.dat"), header=T)
efm$x <- ifelse(efm$growth >= 2, 1,0)
```

7.6a

Compute the difference in incumbent party's vote share on average, comparing those two groups of elections, and determine the standard error for this difference.

```
c1 <- subset(efm, efm$x == 1)
c2 <- subset(efm, efm$x == 0)
diff_mean <- abs(mean(c1$vote) - mean(c2$vote))
std_error_c1 <- function(x) sd(c1$vote)/sqrt(length(c2$vote))
std_error_c2 <- function(x) sd(c2$vote)/sqrt(length(c2$vote))
se_76a <- sqrt(std_error_c1()^2 + std_error_c2()^2)
diff_mean
```

```
## [1] 5.5075
```

```
se_76a
```

```
## [1] 2.502052
```

The difference in incumbent party's vote share on average is 5.5. The standard error for this difference is 2.5.

7.6b

Regress incumbent party's vote share on the binary predictor of income growth and check that the resulting estimate and standard error are the same as above.

```
fit_efm <- lm(efm$vote ~ efm$x)
fit_efm
```

```
##
## Call:
## lm(formula = efm$vote ~ efm$x)
##
## Coefficients:
## (Intercept)        efm$x
##      49.301        5.508
```

```
summary(fit_efm)
```

```
##
## Call:
## lm(formula = efm$vote ~ efm$x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.2087  -3.3706   0.1287   3.3037   6.9812
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   49.301      1.769  27.866 1.15e-13 ***
## efm$x          5.508      2.502   2.201    0.045 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.004 on 14 degrees of freedom
## Multiple R-squared:  0.2571, Adjusted R-squared:  0.204
## F-statistic: 4.845 on 1 and 14 DF,  p-value: 0.045
```

After regressed incumbent party's vote share on the binary predictor of income growth. I can find that the resulting estimate is 5.5 and the standard error is 2.5, they are same as above.

8.8 Comparing lm and stan_glm:

Use simulated data to compare least squares estimation to default Bayesian regression:

8.8a

Simulate 100 data points from the model, y = 2 + 3x + error, with predictors x drawn from a uniform distribution from 0 to 20, and with independent errors drawn from the normal distribution with mean 0 and standard deviation 5. Fit the regression of y on x data using lm and stan_glm (using its default settings) and check that the two programs give nearly identical results.

```
x_88a <- runif(100,0,20)
error_88a <- rnorm(100,0,5)
y_88a <- 2 + 3*x_88a + error_88a
data88c_1 <- data.frame(y_88a, x_88a)
```

```r
fit_88a1 <- lm(y_88a ~ x_88a)
fit_88a2 <- stan_glm(y_88a ~ x_88a)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.059781 seconds (Warm-up)
## Chain 1:                0.057721 seconds (Sampling)
## Chain 1:                0.117502 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.5e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
```

```
## Chain 2:  Elapsed Time: 0.064941 seconds (Warm-up)
## Chain 2:                0.054763 seconds (Sampling)
## Chain 2:                0.119704 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.3e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.059066 seconds (Warm-up)
## Chain 3:                0.077518 seconds (Sampling)
## Chain 3:                0.136584 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.6e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.08603 seconds (Warm-up)
## Chain 4:                0.07318 seconds (Sampling)
## Chain 4:                0.15921 seconds (Total)
## Chain 4:
```

```r
fit_88a1
```

```
##
## Call:
## lm(formula = y_88a ~ x_88a)
##
## Coefficients:
## (Intercept)        x_88a
##       2.385        2.997
```

```r
fit_88a2
```

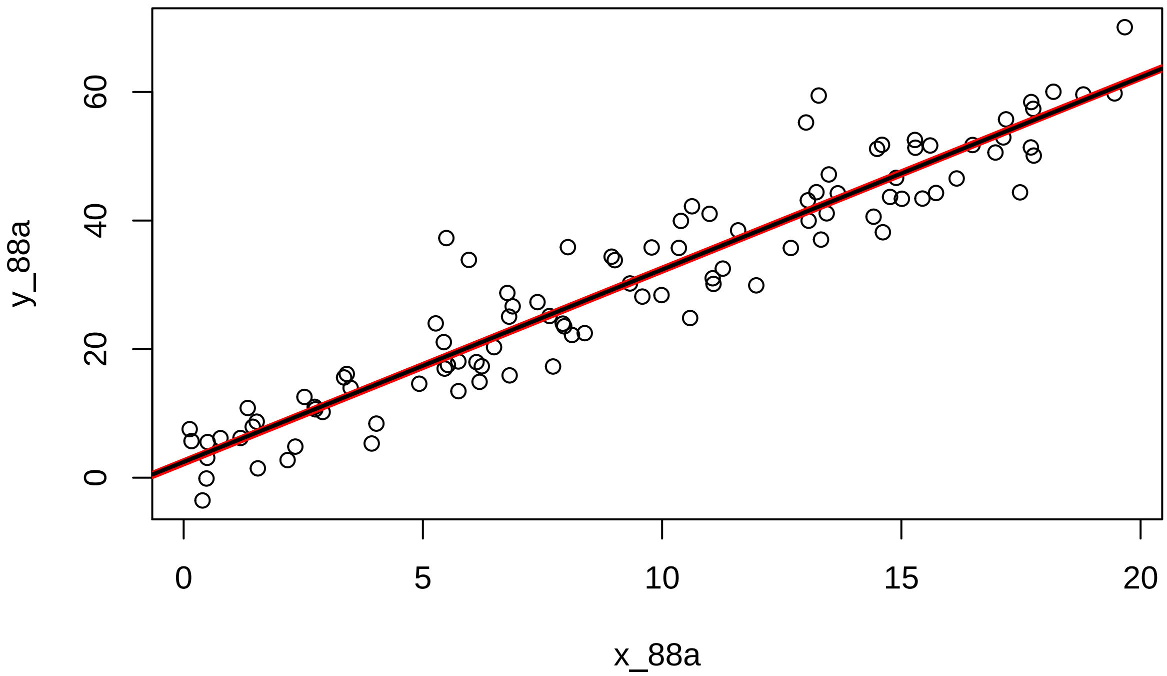
```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_88a ~ x_88a
##  observations: 100
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) 2.4    1.1
## x_88a       3.0    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 5.5    0.4
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

After fitted the regression y on x data using lm and stan_glm, I can get that the two programs give nearly identical results.

8.8b

Plot the simulated data and the two fitted regression lines.

```r
plot(x_88a,y_88a)
abline(fit_88a1, col = "red", lwd = 4)
abline(fit_88a2, col = "black", lwd = 2)
```



The two fitted regression line nearly the same.

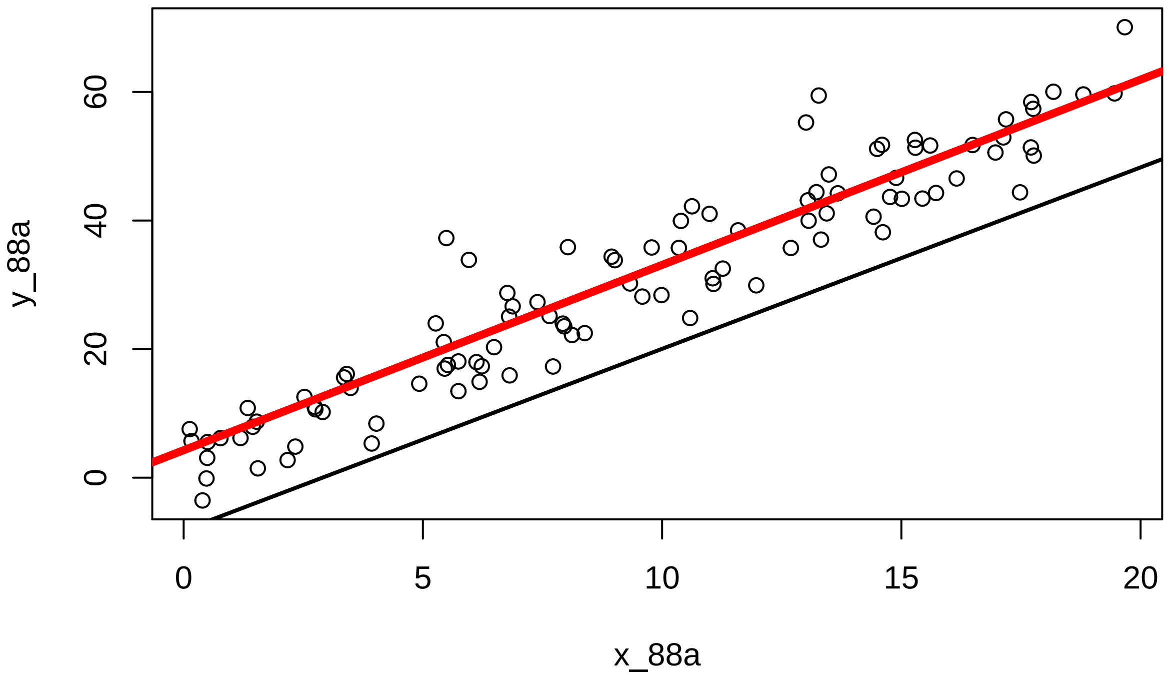
8.8c

Repeat the two steps above, but try to create conditions for your simulation so that lm and stan_glm give much different results.

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.9e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: No variance estimation is
## Chain 1:          performed for num_warmup < 20
## Chain 1:
## Chain 1: Iteration:  1 / 35 [  2%]  (Warmup)
## Chain 1: Iteration:  3 / 35 [  8%]  (Warmup)
## Chain 1: Iteration:  6 / 35 [ 17%]  (Warmup)
## Chain 1: Iteration:  9 / 35 [ 25%]  (Warmup)
## Chain 1: Iteration: 12 / 35 [ 34%]  (Warmup)
## Chain 1: Iteration: 15 / 35 [ 42%]  (Warmup)
## Chain 1: Iteration: 18 / 35 [ 51%]  (Sampling)
## Chain 1: Iteration: 20 / 35 [ 57%]  (Sampling)
## Chain 1: Iteration: 23 / 35 [ 65%]  (Sampling)
## Chain 1: Iteration: 26 / 35 [ 74%]  (Sampling)
## Chain 1: Iteration: 29 / 35 [ 82%]  (Sampling)
## Chain 1: Iteration: 32 / 35 [ 91%]  (Sampling)
## Chain 1: Iteration: 35 / 35 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.001795 seconds (Warm-up)
```

```
## Chain 1:                0.001869 seconds (Sampling)
## Chain 1:                0.003664 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.7e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.17 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: WARNING: No variance estimation is
## Chain 2:          performed for num_warmup < 20
## Chain 2:
## Chain 2: Iteration:  1 / 35 [  2%]  (Warmup)
## Chain 2: Iteration:  3 / 35 [  8%]  (Warmup)
## Chain 2: Iteration:  6 / 35 [ 17%]  (Warmup)
## Chain 2: Iteration:  9 / 35 [ 25%]  (Warmup)
## Chain 2: Iteration: 12 / 35 [ 34%]  (Warmup)
## Chain 2: Iteration: 15 / 35 [ 42%]  (Warmup)
## Chain 2: Iteration: 18 / 35 [ 51%]  (Sampling)
## Chain 2: Iteration: 20 / 35 [ 57%]  (Sampling)
## Chain 2: Iteration: 23 / 35 [ 65%]  (Sampling)
## Chain 2: Iteration: 26 / 35 [ 74%]  (Sampling)
## Chain 2: Iteration: 29 / 35 [ 82%]  (Sampling)
## Chain 2: Iteration: 32 / 35 [ 91%]  (Sampling)
## Chain 2: Iteration: 35 / 35 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.001485 seconds (Warm-up)
## Chain 2:                0.00224 seconds (Sampling)
## Chain 2:                0.003725 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: WARNING: No variance estimation is
## Chain 3:          performed for num_warmup < 20
## Chain 3:
## Chain 3: Iteration:  1 / 35 [  2%]  (Warmup)
## Chain 3: Iteration:  3 / 35 [  8%]  (Warmup)
## Chain 3: Iteration:  6 / 35 [ 17%]  (Warmup)
## Chain 3: Iteration:  9 / 35 [ 25%]  (Warmup)
## Chain 3: Iteration: 12 / 35 [ 34%]  (Warmup)
## Chain 3: Iteration: 15 / 35 [ 42%]  (Warmup)
## Chain 3: Iteration: 18 / 35 [ 51%]  (Sampling)
## Chain 3: Iteration: 20 / 35 [ 57%]  (Sampling)
## Chain 3: Iteration: 23 / 35 [ 65%]  (Sampling)
## Chain 3: Iteration: 26 / 35 [ 74%]  (Sampling)
## Chain 3: Iteration: 29 / 35 [ 82%]  (Sampling)
```

```
## Chain 3: Iteration: 32 / 35 [ 91%]  (Sampling)
## Chain 3: Iteration: 35 / 35 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.001734 seconds (Warm-up)
## Chain 3:                0.002239 seconds (Sampling)
## Chain 3:                0.003973 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.2 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: WARNING: No variance estimation is
## Chain 4:          performed for num_warmup < 20
## Chain 4:
## Chain 4: Iteration:  1 / 35 [  2%]  (Warmup)
## Chain 4: Iteration:  3 / 35 [  8%]  (Warmup)
## Chain 4: Iteration:  6 / 35 [ 17%]  (Warmup)
## Chain 4: Iteration:  9 / 35 [ 25%]  (Warmup)
## Chain 4: Iteration: 12 / 35 [ 34%]  (Warmup)
## Chain 4: Iteration: 15 / 35 [ 42%]  (Warmup)
## Chain 4: Iteration: 18 / 35 [ 51%]  (Sampling)
## Chain 4: Iteration: 20 / 35 [ 57%]  (Sampling)
## Chain 4: Iteration: 23 / 35 [ 65%]  (Sampling)
## Chain 4: Iteration: 26 / 35 [ 74%]  (Sampling)
## Chain 4: Iteration: 29 / 35 [ 82%]  (Sampling)
## Chain 4: Iteration: 32 / 35 [ 91%]  (Sampling)
## Chain 4: Iteration: 35 / 35 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.003075 seconds (Warm-up)
## Chain 4:                0.02624 seconds (Sampling)
## Chain 4:                0.029315 seconds (Total)
## Chain 4:

## Warning: There were 4 chains where the estimated Bayesian Fraction of Missing Information was low. S
## http://mc-stan.org/misc/warnings.html#bfmi-low

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

## Warning: The largest R-hat is 1.87, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#r-hat

## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess

## Warning: Markov chains did not converge! Do not analyze results!
```




```
##
## Call:
## lm(formula = y_88a ~ x_88a, data = data88c_2)
##
## Coefficients:
## (Intercept)        x_88a
##       4.270        2.883
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_88a ~ x_88a
##  observations: 101
##  predictors:   2
## ------
##             Median MAD_SD
## (Intercept) -8.2    9.9
## x_88a        2.8    0.4
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 16.4    8.4
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

Add an outlier for the original dataframe, we can find that two regression lines are much different from each other.

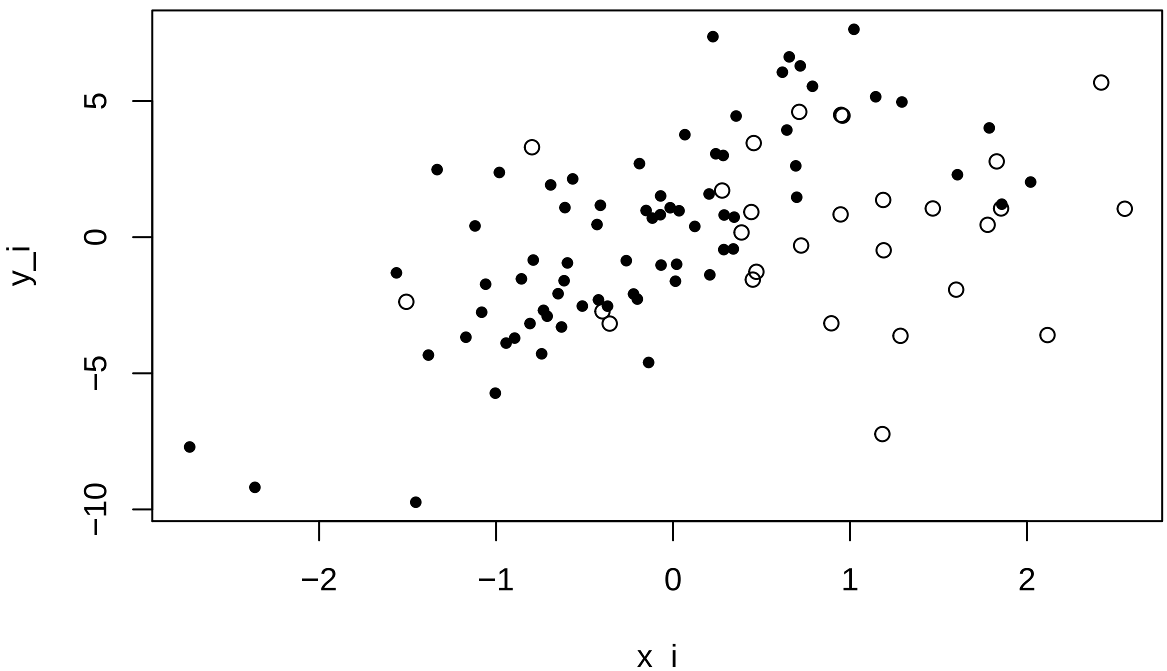
10.1 Regression with interactions:

Simulate 100 data points from the model, y = b0 + b1 x + b2 z + b3 xz + error, with a continuous predictor x and a binary predictor z, coefficients b = c(1, 2, -1, -2), and errors drawn independently from a normal distribution with mean 0 and standard deviation 3, as follows. For each data point i, first draw zi, equally likely to take on the values 0 and 1. Then draw xi from a normal distribution with mean zi and standard deviation 1. Then draw the error from its normal distribution and compute yi.

10.1a

Display your simulated data as a graph of y vs. x, using dots and circles for the points with z = 0 and 1, respectively.

```
error <- rnorm(100,0,3)
z_i <- rbinom(100, c(0,1), 0.5)
x_i <- rnorm(100, z_i,1)
y_i = 1 + 2*x_i - 1*z_i -2*x_i*z_i + error
shape <- ifelse(z_i==1, 1,20)
plot(x_i,y_i,pch = shape)
```



10.1b

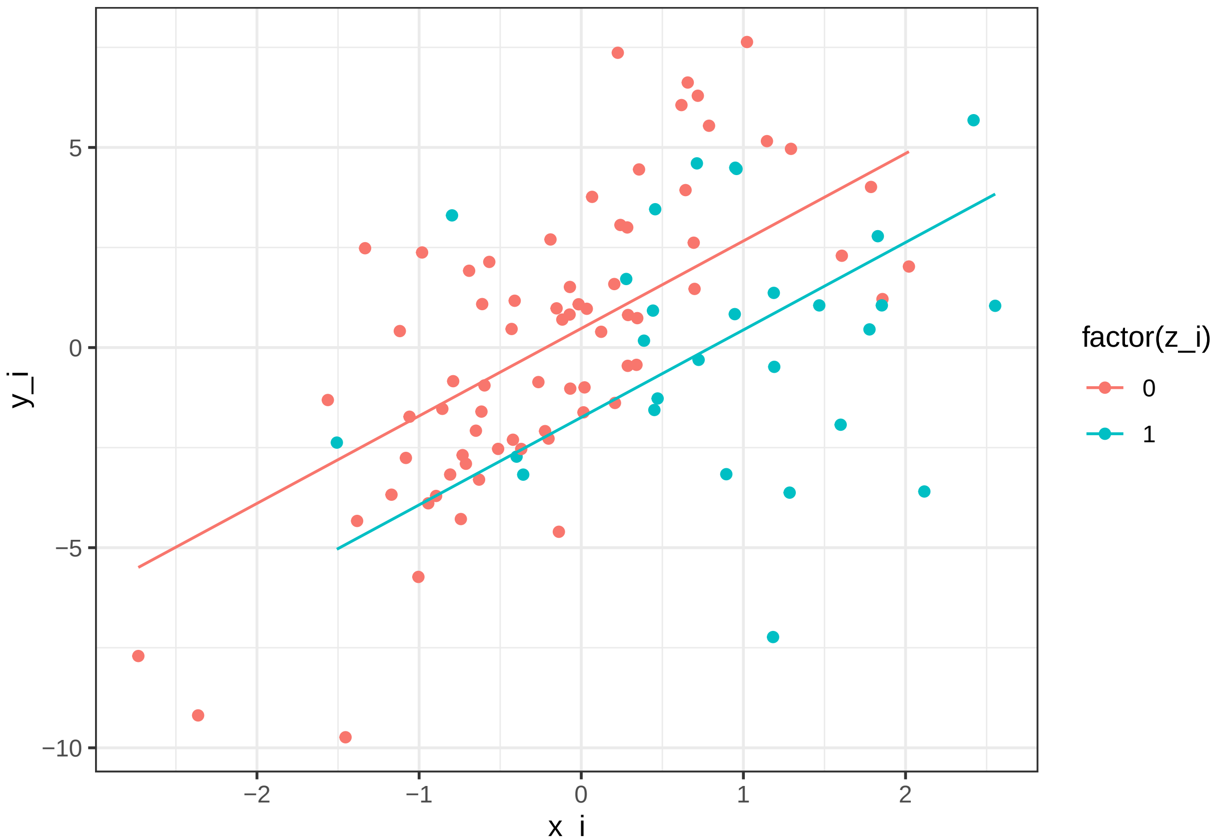
Fit a regression predicting y from x and z with no interaction. Make a graph with the data and two parallel lines showing the fitted model.

```
library("broom")
```

```
## Warning: package 'broom' was built under R version 3.6.2
```

```
datay_i <- data.frame(x_i,y_i,z_i)
fitmod_101b <- lm(formula = y_i ~ x_i+z_i, data = datay_i)
augmented_mod_101b <- augment(fitmod_101b)
```

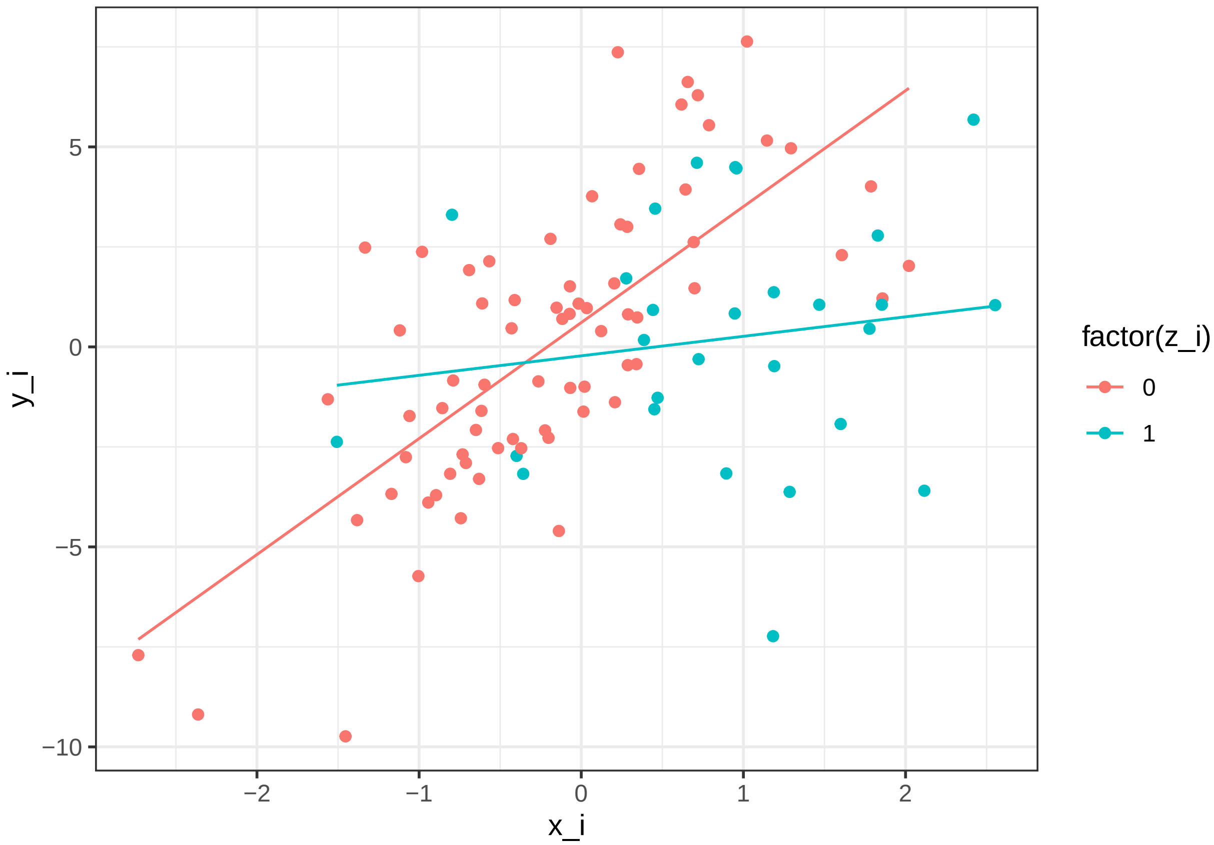
```r
ggplot(data = augmented_mod_101b, aes(x = x_i, y = y_i, color = factor(z_i))) +
  geom_point()+
  geom_line(aes(x = x_i, y = .fitted)) +
  theme_bw()
```



10.1c

Fit a regression predicting y from x, z, and their interaction. Make a graph with the data and two lines showing the fitted model.

```r
library("broom")
fitmod_101c <- lm(y_i ~ x_i + z_i + x_i:z_i)
augmented_mod_101c <- augment(fitmod_101c)
ggplot(augmented_mod_101c, aes(y = y_i, x = x_i,color = factor(z_i))) +
  geom_point() +
  geom_line(aes(x = x_i, y = .fitted)) +
  theme_bw()
```



10.2 Regression with interactions:

Here is the output from a fitted linear regression of outcome y on pre-treatment predictor x, treatment indicator z, and their interaction:

10.2a

Write the equation of the estimated regression line of y on x for the treatment group and the control group, and the equation of the estimated regression line of y on x for the control group.

For the treatment group$(z = 1)$ $y = 2.3 * x + 3.9$ For the control group$(z = 0)$ $y = 1.6 * x + 1.2$

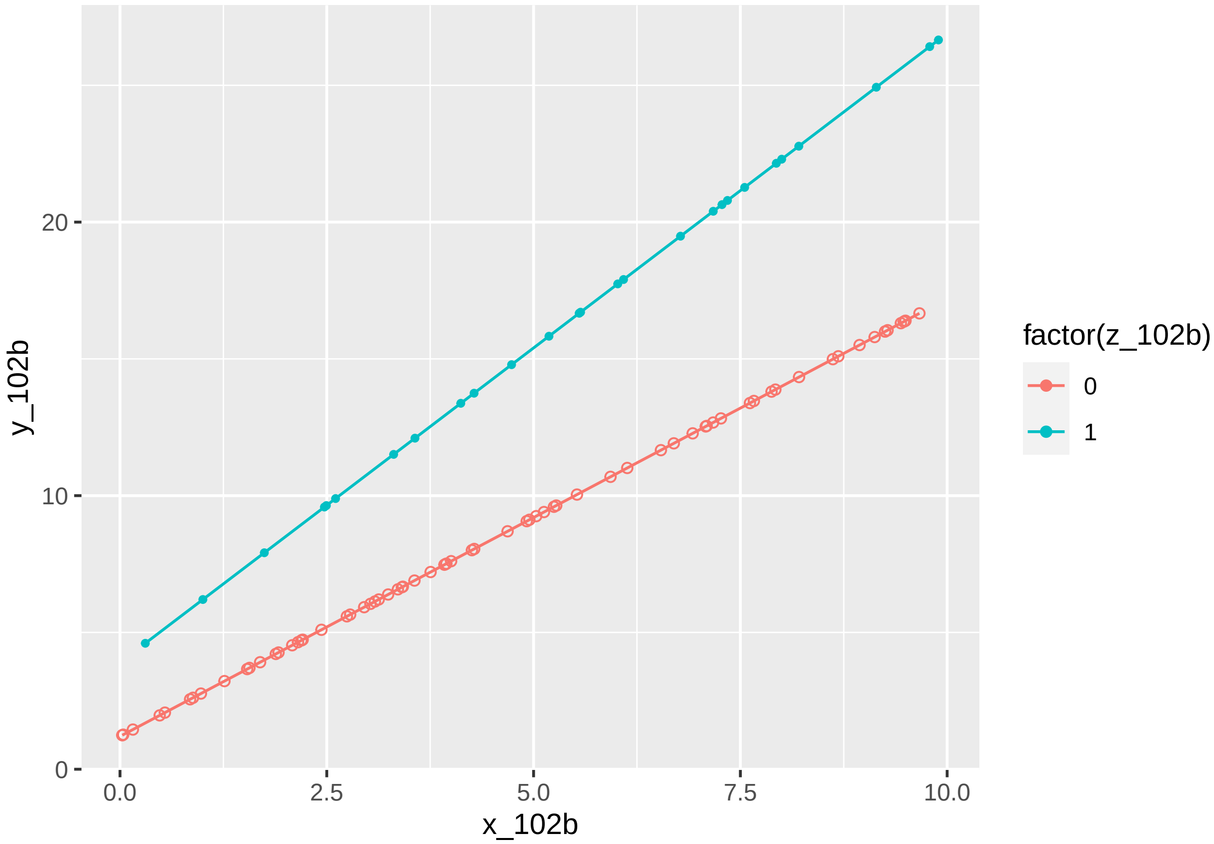
10.2b

Graph with pen on paper the two regression lines, assuming the values of x fall in the range (0, 10). On this graph also include a scatterplot of data (using open circles for treated units and dots for controls) that are consistent with the fitted model.

set total 100 simulation data points.

```
library("broom")
x_102b <- runif(100,0,10)
z_102b <- rbinom(100, c(0,1), 0.5)
y_102b <- 1.2 + 1.6*x_102b + 2.7*z_102b + 0.7*x_102b*z_102b
data_102b <- data.frame(y_102b, x_102b, z_102b)
fit_102b <- lm(y_102b ~ x_102b + z_102b + x_102b:z_102b)
```

```r
augmented_mod_102b <- augment(fit_102b)
shape_102b <- ifelse(z_102b==0, 1,20)
data_space_102b <- ggplot(data = augmented_mod_102b, aes(x = x_102b, y = y_102b, color = factor(z_102b))
  geom_point(pch = shape_102b)+
  geom_line(aes(x = x_102b, y = .fitted))
data_space_102b
```



10.5 Regression modeling and prediction:

The folder KidIQ contains a subset of the children and mother data discussed earlier in the chapter. You have access to children's test scores at age 3, mother's education, and the mother's age at the time she gave birth for a sample of 400 children.

```r
kidiq_105 <- read.csv("https://raw.githubusercontent.com/avehtari/ROS-Examples/master/KidIQ/data/child_
```

10.5a

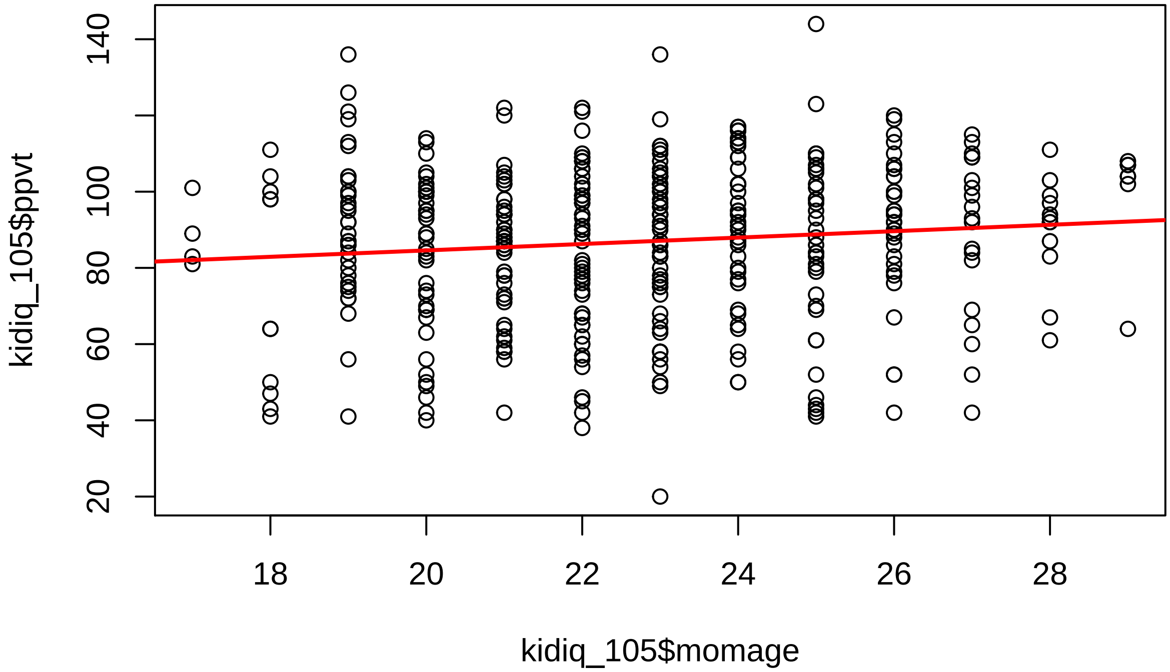
Fit a regression of child test scores on mother's age, display the data and fitted model, check assumptions, and interpret the slope coefficient. Based on this analysis, when do you recommend mothers should give birth? What are you assuming in making this recommendation?

```r
fit_105a <- lm(kidiq_105$ppvt ~ kidiq_105$momage)
fit_105a
```

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

```
## lm(formula = kidiq_105$ppvt ~ kidiq_105$momage)
##
## Coefficients:
##      (Intercept)  kidiq_105$momage
##          67.7827            0.8403
```

```
plot(x = kidiq_105$momage, y = kidiq_105$ppvt)
abline(fit_105a, col = "red",lwd = 2)
```



According to the data and fitted model,the intercept coefficient is 67.78, slope coefficient is 0.84. And based on the anaylsis, I recommend that mothers shoudl give birth not before age 26. Because the slope is positive, which means that as the age is getting bigger, the test score is getting bigger as well.

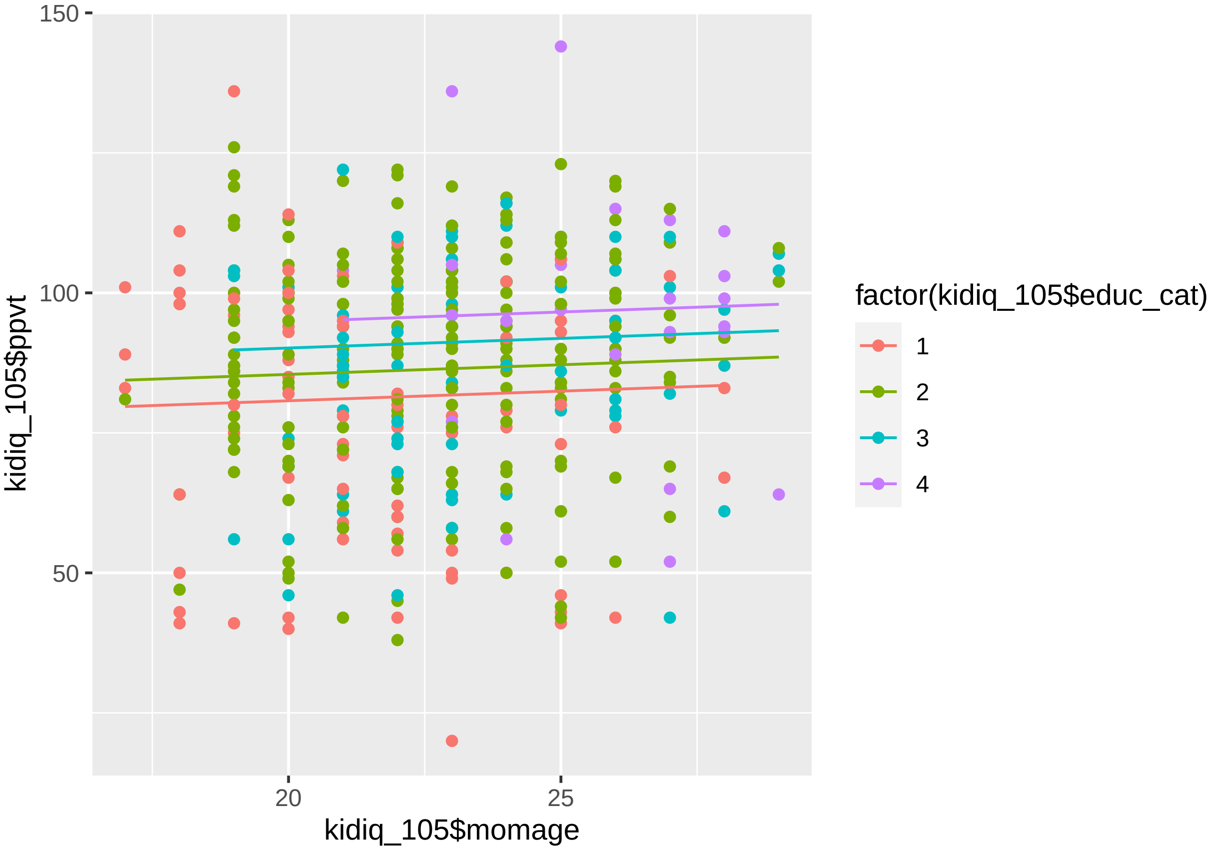
10.5b

Repeat this for a regression that further includes mother's education, interpreting both slope coefficients in this model. Have your conclusions about the timing of birth changed?

```
library("broom")
fit_105b <- lm(kidiq_105$ppvt ~ kidiq_105$momage + kidiq_105$educ_cat)
augmented_mod_105b <- augment(fit_105b)
data_space_105b <- ggplot(data = augmented_mod_105b, aes(x = kidiq_105$momage, y = kidiq_105$ppvt, colo
  geom_point()+
  geom_line(aes(x = kidiq_105$momage, y = .fitted))
fit_105b
```

```
##
## Call:
## lm(formula = kidiq_105$ppvt ~ kidiq_105$momage + kidiq_105$educ_cat)
##
## Coefficients:
##        (Intercept)    kidiq_105$momage  kidiq_105$educ_cat
##            69.1554              0.3433              4.7114
```

```
data_space_105b
```



After repeating this for a regression that further includes mother's education, the intercept coefficient is 69.16, the slope coefficient is 0.34 for mom's age and 4.71 for mom's education. Based on the slope coefficient are both positive, my conclusion about the timing of birth remain same.

10.5c

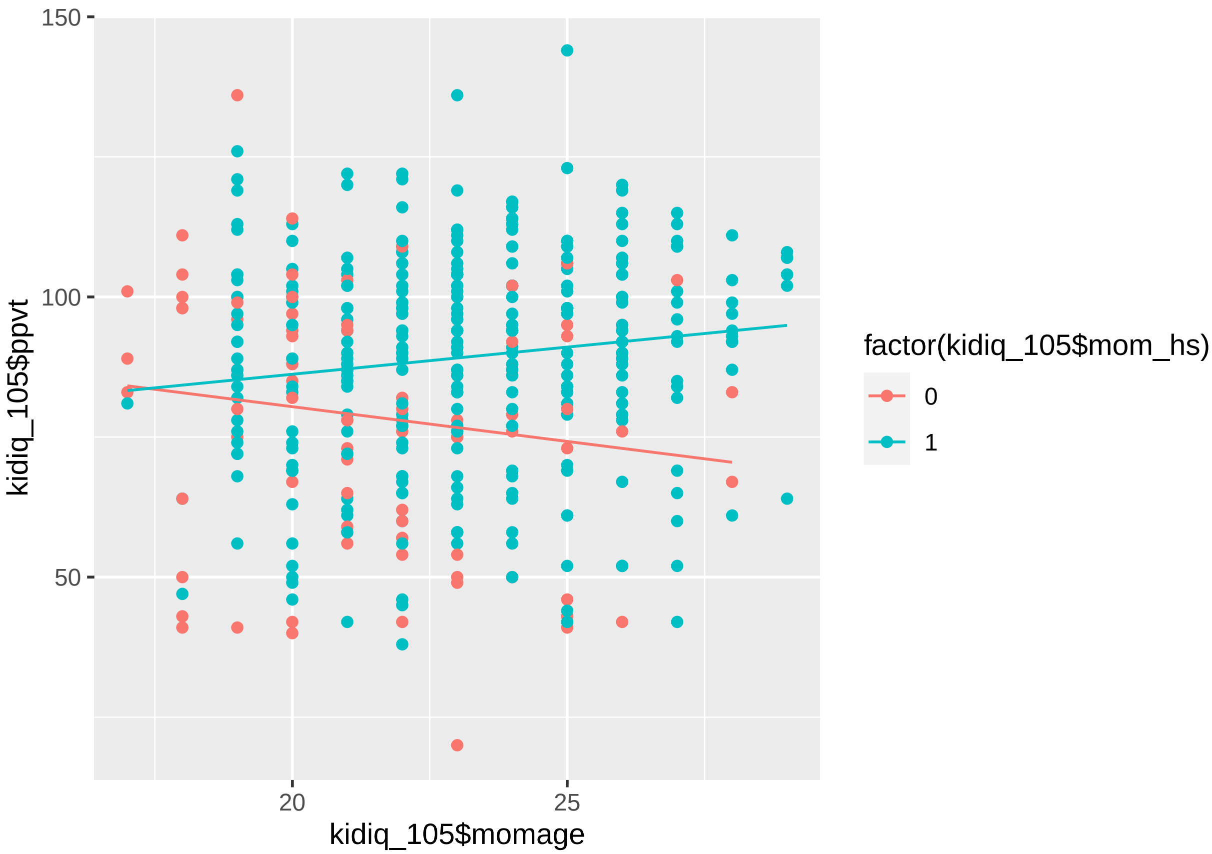
Now create an indicator variable reflecting whether the mother has completed high school or not. Consider interactions between high school completion and mother's age. Also create a plot that shows the separate regression lines for each high school completion status group.

```
#To create an indicator variable for high school completion
kidiq_105$mom_hs <- ifelse(kidiq_105$educ_cat >= 2, 1,0)
library("broom")
fit_105c <- lm(kidiq_105$ppvt ~ kidiq_105$momage + kidiq_105$mom_hs + kidiq_105$momage:kidiq_105$mom_hs)
augmented_mod_105c <- augment(fit_105c)
data_space_105c <- ggplot(data = augmented_mod_105c, aes(x = kidiq_105$momage, y = kidiq_105$ppvt, colo
  geom_point()+
  geom_line(aes(x = kidiq_105$momage, y = .fitted))
fit_105c
```

```
##
## Call:
## lm(formula = kidiq_105$ppvt ~ kidiq_105$momage + kidiq_105$mom_hs +
##     kidiq_105$momage:kidiq_105$mom_hs)
##
```

```
## Coefficients:
##                    (Intercept)                   kidiq_105$momage
##                         105.22                              -1.24
##               kidiq_105$mom_hs  kidiq_105$momage:kidiq_105$mom_hs
##                         -38.41                               2.21
```

```
data_space_105c
```



10.5d

Finally, fit a regression of child test scores on mother's age and education level for the first 200 children and use this model to predict test scores for the next 200. Graphically display comparisons of the predicted and actual scores for the final 200 children.

```
library(modelbased)
```

```
## Registered S3 methods overwritten by 'parameters':
##   method                           from
##   as.double.parameters_kurtosis    datawizard
##   as.double.parameters_skewness    datawizard
##   as.double.parameters_smoothness  datawizard
##   as.numeric.parameters_kurtosis   datawizard
##   as.numeric.parameters_skewness   datawizard
##   as.numeric.parameters_smoothness datawizard
##   print.parameters_distribution    datawizard
##   print.parameters_kurtosis        datawizard
##   print.parameters_skewness        datawizard
```

```
##   summary.parameters_kurtosis      datawizard
##   summary.parameters_skewness      datawizard

## Registered S3 method overwritten by 'modelbased':
##   method                     from
##   print.visualisation_recipe datawizard
```

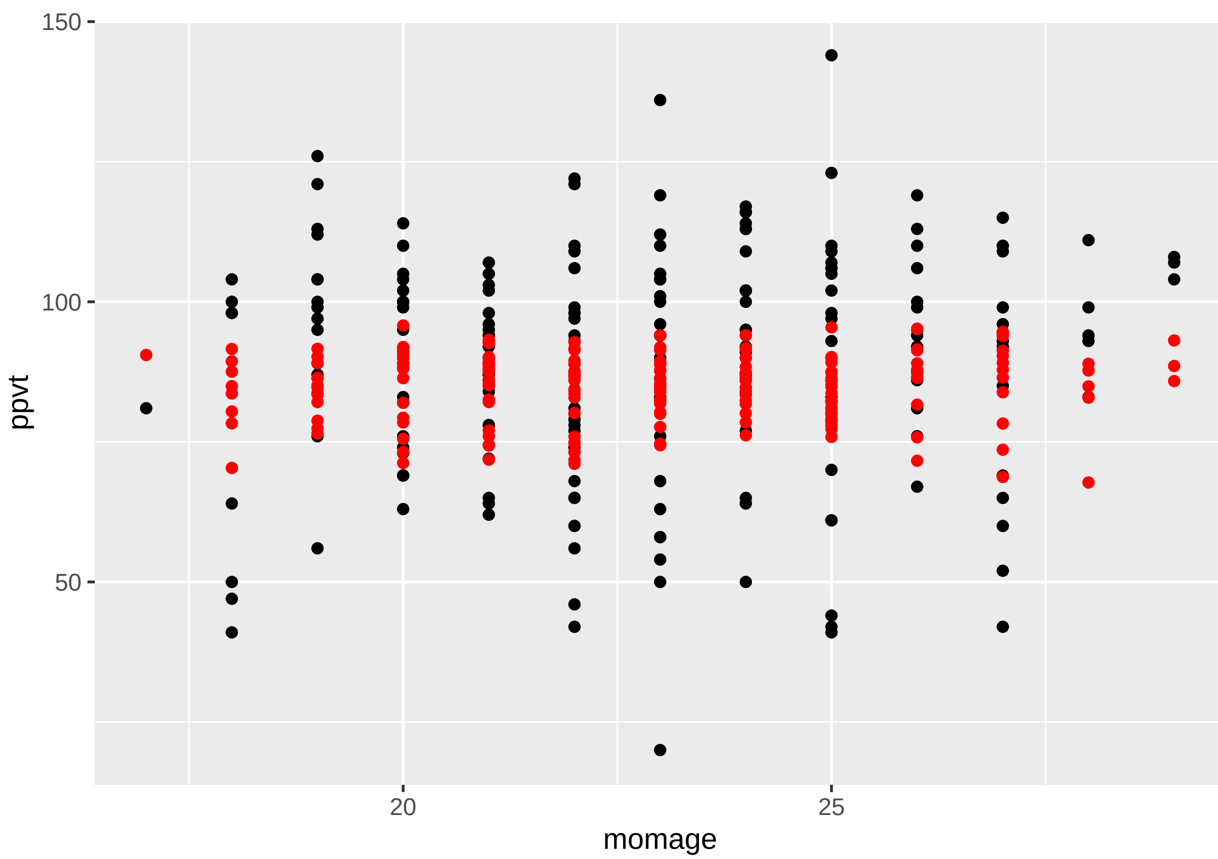
```
data105d <- head(kidiq_105, n = 200)
actual_sc <- tail(kidiq_105, n = 200)
kid_score <- data105d$ppvt
mom_age <- data105d$momage
mom_hs_105d <- data105d$mom_hs
fit_105d <- stan_glm(kid_score ~ mom_age + mom_hs_105d + mom_age:mom_hs_105d,data = data105d)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.2 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.373519 seconds (Warm-up)
## Chain 1:                0.400999 seconds (Sampling)
## Chain 1:                0.774518 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
```

```
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.320325 seconds (Warm-up)
## Chain 2:                0.323314 seconds (Sampling)
## Chain 2:                0.643639 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.5e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.322672 seconds (Warm-up)
## Chain 3:                0.318403 seconds (Sampling)
## Chain 3:                0.641075 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2.1e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
```

```
## Chain 4:
## Chain 4:  Elapsed Time: 0.334945 seconds (Warm-up)
## Chain 4:                0.352695 seconds (Sampling)
## Chain 4:                0.68764 seconds (Total)
## Chain 4:
```

```
postpred <- posterior_predict(fit_105d, newdata = actual_sc)
pred_data <- apply(postpred, 2, mean)
pred_data <- data.frame(pred_data)
pred_data$educ_cat <- actual_sc$educ_cat
pred_data$momage <- actual_sc$momage
pred_data$mom_hs <- actual_sc$mom_hs
colnames(pred_data)[1] <- "ppvt"
plot_105d <- ggplot(actual_sc, aes(x = momage, y = ppvt)) +
    geom_point() +
    geom_point(data = pred_data, col = "red")
plot_105d
```



The dots in the color of red are predicted value for the last 200 children. The dots in color black is the original data.

10.6 Regression models with interactions:

The folder Beauty contains data (use file beauty.csv) Beauty and teaching evaluations from Hamermesh and Parker (2005) on student evaluations of instructors' beauty and teaching quality for several courses at the University of Texas. The teaching evaluations were conducted at the end of the semester, and the beauty judgments were made later, by six students who had not attended the classes and were not aware of the course evaluations.

See also Felton, Mitchell, and Stinson (2003) for more on this topic.

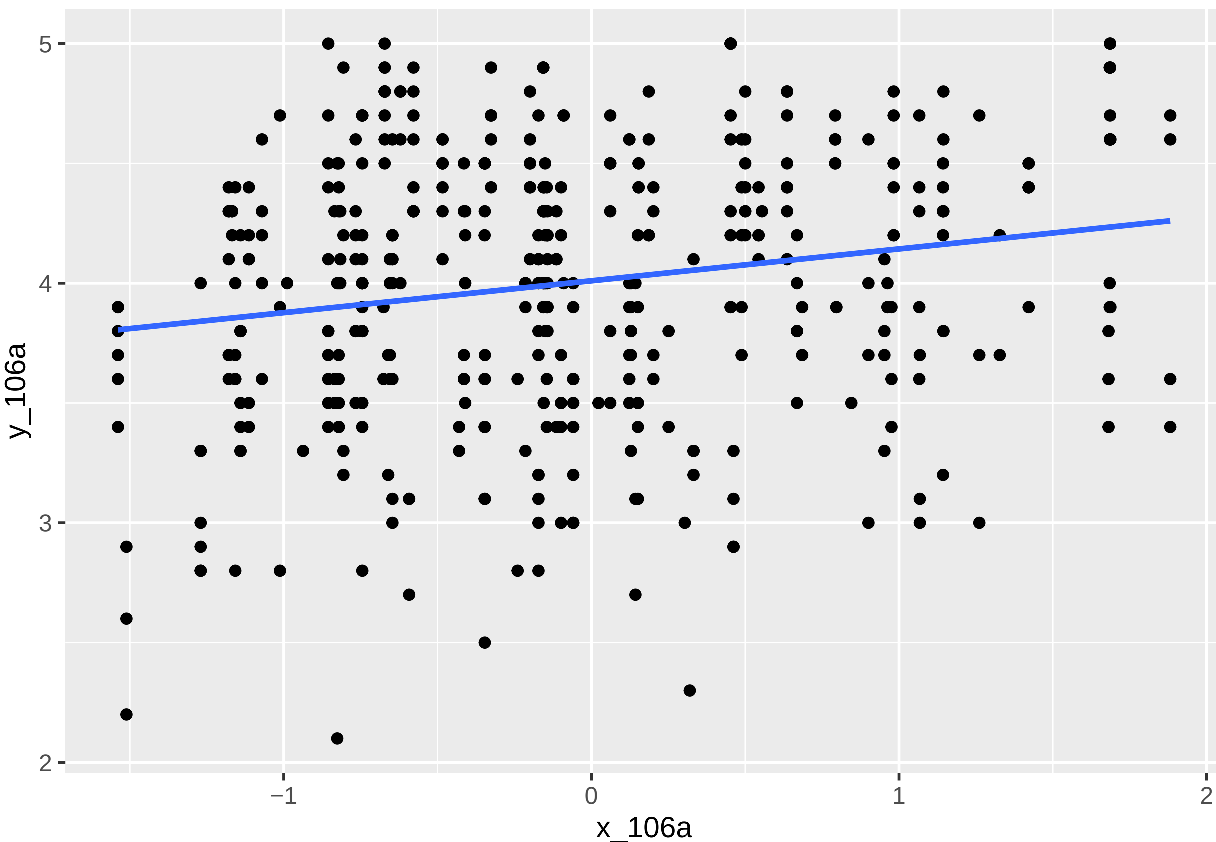
```
beauty <- read.csv("https://raw.githubusercontent.com/avehtari/ROS-Examples/master/Beauty/data/beauty.c
```

10.6a

Run a regression using beauty (the variable beauty) to predict course evaluations (eval), adjusting for various other predictors. Graph the data and fitted model, and explain the meaning of each of the coefficients along with the residual standard deviation. Plot the residuals versus fitted values.

```
x_106a <- beauty$beauty
y_106a <- beauty$eval
fit_106a <- lm(y_106a ~ x_106a)
data_space_106a <- ggplot(data = beauty, aes(x = x_106a, y = y_106a))+
  geom_point()+
  geom_smooth(method = "lm", se = FALSE)
data_space_106a
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
summary(fit_106a)
```

```
##
## Call:
## lm(formula = y_106a ~ x_106a)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.80015 -0.36304  0.07254  0.40207  1.10373
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.01002    0.02551 157.205  < 2e-16 ***
## x_106a       0.13300    0.03218   4.133 4.25e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5455 on 461 degrees of freedom
## Multiple R-squared:  0.03574,    Adjusted R-squared:  0.03364
## F-statistic: 17.08 on 1 and 461 DF,  p-value: 4.247e-05
```

```
sd(x_106a)
```

```
## [1] 0.7886476
```

As we can see from the summary of the fitted regression model. The intercept is 4.01002, the slope for beauty is 0.13300, and the standard deviation for the residual is 0.7886476

10.6b

Fit some other models, including beauty and also other predictors. Consider at least one model with interactions. For each model, explain the meaning of each of its estimated coefficients.

```
#add age into the model and re-fit the model
x_age <- beauty$age
fit_106b_1 <- stan_glm(y_106a ~ x_106a+x_age)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
```

```
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.065388 seconds (Warm-up)
## Chain 1:                0.10166 seconds (Sampling)
## Chain 1:                0.167048 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.058238 seconds (Warm-up)
## Chain 2:                0.088765 seconds (Sampling)
## Chain 2:                0.147003 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.5e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.057049 seconds (Warm-up)
## Chain 3:                0.083982 seconds (Sampling)
```

```
## Chain 3:                0.141031 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.4e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.14 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.054682 seconds (Warm-up)
## Chain 4:                0.098189 seconds (Sampling)
## Chain 4:                0.152871 seconds (Total)
## Chain 4:
```

```
summary(fit_106b_1)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   3
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) 4.0    0.1  3.8   4.0   4.2
## x_106a      0.1    0.0  0.1   0.1   0.2
## x_age       0.0    0.0  0.0   0.0   0.0
## sigma       0.5    0.0  0.5   0.5   0.6
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
```

```
##                mcse Rhat n_eff
## (Intercept)   0.0  1.0  4269
## x_106a        0.0  1.0  4684
## x_age         0.0  1.0  4305
## sigma         0.0  1.0  4778
## mean_PPD      0.0  1.0  4098
## log-posterior 0.0  1.0  1866
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
#add if gender as a binary predictor and re-fit the model
g_106b <- beauty$female
fit_106b_2 <- stan_glm(y_106a ~ x_106a+x_age+g_106b)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 8.6e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.86 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.075783 seconds (Warm-up)
## Chain 1:                0.089851 seconds (Sampling)
## Chain 1:                0.165634 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
```

```
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.058706 seconds (Warm-up)
## Chain 2:                0.104697 seconds (Sampling)
## Chain 2:                0.163403 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.6e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.059545 seconds (Warm-up)
## Chain 3:                0.0903 seconds (Sampling)
## Chain 3:                0.149845 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.7e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.17 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
```

```
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.060251 seconds (Warm-up)
## Chain 4:                0.09063 seconds (Sampling)
## Chain 4:                0.150881 seconds (Total)
## Chain 4:
```

```
summary(fit_106b_2)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   4
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  4.2    0.1  4.0   4.2   4.4
## x_106a       0.1    0.0  0.1   0.1   0.2
## x_age        0.0    0.0  0.0   0.0   0.0
## g_106b      -0.2    0.1 -0.3  -0.2  -0.1
## sigma        0.5    0.0  0.5   0.5   0.6
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  3812
## x_106a        0.0  1.0  4322
## x_age         0.0  1.0  3995
## g_106b        0.0  1.0  4660
## sigma         0.0  1.0  5635
## mean_PPD      0.0  1.0  4599
## log-posterior 0.0  1.0  1960
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
#add the interaction between beauty and gender into considerartion
fit_106b_3 <- stan_glm(y_106a ~ x_106a + g_106b + x_106a*g_106b)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
```

```
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.

##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.068654 seconds (Warm-up)
## Chain 1:                0.099496 seconds (Sampling)
## Chain 1:                0.16815 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.2e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.069973 seconds (Warm-up)
## Chain 2:                0.095028 seconds (Sampling)
## Chain 2:                0.165001 seconds (Total)
## Chain 2:
##
```

```
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.5e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.25 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.069807 seconds (Warm-up)
## Chain 3:                0.114401 seconds (Sampling)
## Chain 3:                0.184208 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.067997 seconds (Warm-up)
## Chain 4:                0.103786 seconds (Sampling)
## Chain 4:                0.171783 seconds (Total)
## Chain 4:
```

```r
summary(fit_106b_3)
```

```
##
## Model Info:
##  function:     stan_glm
```

```
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + g_106b + x_106a * g_106b
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   4
##
## Estimates:
##                 mean   sd   10%   50%   90%
## (Intercept)    4.1    0.0  4.1   4.1   4.1
## x_106a         0.2    0.0  0.1   0.2   0.3
## g_106b        -0.2    0.1 -0.3  -0.2  -0.1
## x_106a:g_106b -0.1    0.1 -0.2  -0.1   0.0
## sigma          0.5    0.0  0.5   0.5   0.6
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  3512
## x_106a        0.0  1.0  2580
## g_106b        0.0  1.0  3342
## x_106a:g_106b 0.0  1.0  2607
## sigma         0.0  1.0  4139
## mean_PPD      0.0  1.0  3681
## log-posterior 0.0  1.0  1791
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```
#add everything expect lower, minority and course_id into consideration
x_ifeng <- beauty$nonenglish
fit_106b_4 <- stan_glm(y_106a ~ x_106a + x_age + g_106b + x_ifeng)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.

##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
```

```
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.055459 seconds (Warm-up)
## Chain 1:                0.094448 seconds (Sampling)
## Chain 1:                0.149907 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.8e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.18 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.056768 seconds (Warm-up)
## Chain 2:                0.094297 seconds (Sampling)
## Chain 2:                0.151065 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.5e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
```

```
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.06232 seconds (Warm-up)
## Chain 3:                0.091774 seconds (Sampling)
## Chain 3:                0.154094 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.06051 seconds (Warm-up)
## Chain 4:                0.089429 seconds (Sampling)
## Chain 4:                0.149939 seconds (Total)
## Chain 4:
```

```r
summary(fit_106b_4)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b + x_ifeng
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   5
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  4.2    0.1  4.1   4.2   4.4
## x_106a       0.1    0.0  0.1   0.1   0.2
## x_age        0.0    0.0  0.0   0.0   0.0
## g_106b      -0.2    0.1 -0.3  -0.2  -0.1
## x_ifeng     -0.3    0.1 -0.5  -0.3  -0.2
```

```
## sigma        0.5    0.0  0.5   0.5   0.6
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 4.0    0.0  4.0   4.0   4.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.0  1.0  4255
## x_106a        0.0  1.0  5266
## x_age         0.0  1.0  4451
## g_106b        0.0  1.0  4344
## x_ifeng       0.0  1.0  5777
## sigma         0.0  1.0  5529
## mean_PPD      0.0  1.0  4048
## log-posterior 0.0  1.0  1688
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

For the first fitted model. I added age as the second residual to predict the course evaluation. According to the summary, the intercept is 4.0, which means that when the beauty residual and age residual is 0, the evaluation point is 4.0. The slope for the residual beauty is 0.1, and the slope for the residual age is 0, those coefficients mean that when the residual beauty and age increase by 1, the evaluation point will go up by 0.1 and 0. Which means the age predictor will have light affect on the outcome.

For the second fitted model. I added a binary predictor gender into consideration and re-fit the model. According to the summary. The intercept is 4.2, which means that when the beauty residual and age residual is 0 and the instructor is male, the evaluation point is 4.2. The slope for the residual beauty is 0.1, and the slope for the residual age is 0.0, the slope coefficient for gender is -0.2.

For the third fitted model. I add the interaction between beauty and gender into consideration. According to the summary, the intercept is 4.1, which means that when the beauty residual is 0, and the instruction is male, the evaluation point is 4.1. The slope for the residual beauty is 0.2, and the slope for the gender is -0.2. The slope coefficient for the interaction is -0.1.

For the fourth fitted model. According to the summary. The intercept is 4.2, which means that when all the predictors are 0, the course evaluation point will be 4.2. The slope coefficients for beauty, age, gender, whether native english speaker are 0.1, 0.0, -0.2 and -0.3.

10.7 Predictive simulation for linear regression:

Take one of the models from the previous exercise.

10.7a

Instructor A is a 50-year-old woman who is a native English speaker and has a beauty score of -1. Instructor B is a 60-year-old man who is a native English speaker and has a beauty score of -0.5. Simulate 1000 random draws of the course evaluation rating of these two instructors. In your simulation, use posterior_predict to account for the uncertainty in the regression parameters as well as predictive uncertainty.

Let's choose the second model

```r
fit_106b_2 <- stan_glm(y_106a ~ x_106a + x_age + g_106b)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.9e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.064691 seconds (Warm-up)
## Chain 1:                0.100719 seconds (Sampling)
## Chain 1:                0.16541 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.8e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.18 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.059909 seconds (Warm-up)
```

```
## Chain 2:                0.085293 seconds (Sampling)
## Chain 2:                0.145202 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.3e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.057779 seconds (Warm-up)
## Chain 3:                0.08662 seconds (Sampling)
## Chain 3:                0.144399 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.058328 seconds (Warm-up)
## Chain 4:                0.096323 seconds (Sampling)
## Chain 4:                0.154651 seconds (Total)
## Chain 4:
```

```r
pp_A <- posterior_predict(fit_106b_2, beauty = -1, age = 50, female = 1, draws = 1000)
pp_B <- posterior_predict(fit_106b_2, beauty = -0.5, age = 60, female = 0, draws = 1000)
eval_A <- data.frame(apply(pp_A, 2, mean))
eval_B <- data.frame(apply(pp_B, 2, mean))
mean(eval_A$apply.pp_A..2..mean.)
```

```
## [1] 3.999415
```

```r
mean(eval_B$apply.pp_B..2..mean.)
```

```
## [1] 3.997148
```

The simulated course evaluation points for both instructor A and B are nearly the same.

10.7b

Make a histogram of the difference between the course evaluations for A and B. What is the probability that A will have a higher evaluation?

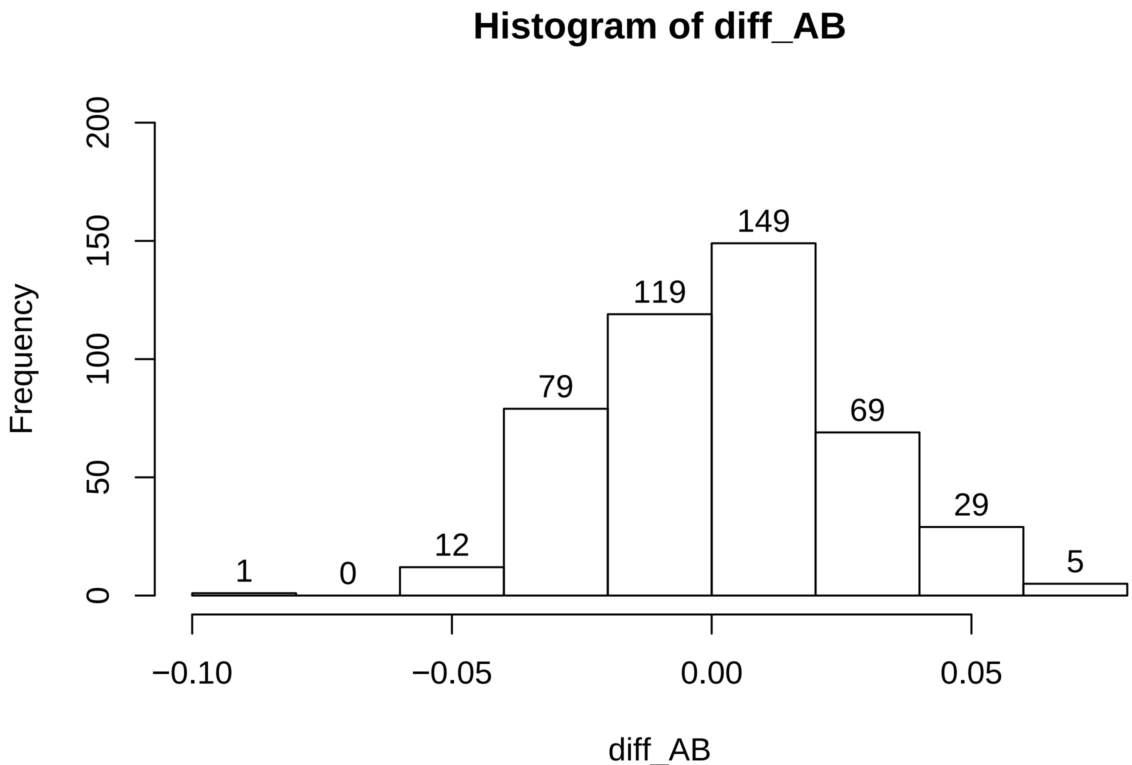
```r
i <- c(1:463)
x_107b1 <- eval_A$apply.pp_A..2..mean.
x_107b2 <- eval_B$apply.pp_B..2..mean.
diff_AB <- x_107b1[i] - x_107b2[i]
diff_AB
```

```
##   [1]  1.471121e-02 -7.047081e-03  3.434726e-02  6.291684e-03 -2.978378e-02
##   [6] -2.009854e-02 -8.071733e-03  4.599399e-02 -2.446265e-02  3.367435e-02
##  [11]  2.074968e-02  2.471569e-03 -9.265041e-03  2.795032e-02 -3.197033e-02
##  [16]  3.194359e-02  9.775229e-03 -1.716156e-03  4.783779e-02 -4.042642e-02
##  [21]  5.122275e-03 -1.197076e-02  3.660469e-02  1.997251e-02  9.953153e-03
##  [26]  1.607548e-02 -3.918011e-02 -3.332340e-02 -3.968079e-02  1.753053e-02
##  [31] -1.390422e-03  2.214100e-03  1.648479e-03  8.690846e-03 -7.227155e-03
##  [36] -3.557828e-02 -2.518896e-02  4.537513e-02 -1.816316e-02  4.253563e-03
##  [41] -1.865479e-02  5.342222e-03  1.136874e-02  1.346390e-03 -1.696852e-02
##  [46]  4.656929e-02 -3.447283e-02  1.496510e-02  9.913763e-04  2.377532e-02
##  [51]  1.753652e-02  2.913558e-02  5.365453e-03 -2.300786e-03 -4.950803e-03
##  [56]  4.720000e-02  1.228400e-02 -2.286182e-02  5.002157e-02  2.358443e-02
##  [61]  3.864140e-02 -9.480608e-03  4.442851e-02  1.485069e-02 -3.983132e-03
##  [66]  4.501115e-02 -1.245484e-02  1.925864e-02 -1.742164e-02  3.002553e-02
##  [71]  1.523232e-02  1.128263e-03  2.362080e-02 -2.675813e-02  7.439364e-02
##  [76] -1.777014e-02  3.390207e-02  2.774473e-02  1.157792e-02  1.432541e-02
##  [81] -1.026382e-02  3.010776e-02  1.823605e-02 -2.632635e-03  1.382629e-02
##  [86] -7.353120e-03  3.703973e-02  2.310758e-02  3.260536e-02 -4.257526e-02
##  [91] -2.108595e-02  2.748485e-03  2.169009e-02  1.574701e-02 -5.469112e-03
##  [96] -1.390581e-02 -3.125938e-02 -3.030191e-02 -3.244324e-02 -2.049835e-02
## [101] -1.851705e-02 -4.652381e-03 -3.620510e-02 -3.382628e-02  2.201618e-02
## [106] -2.204085e-02  4.664647e-02  2.196529e-02  2.222935e-02  2.094035e-02
## [111] -3.409923e-02  1.216898e-02 -6.881763e-03  1.487772e-02  3.610193e-02
## [116]  7.322534e-03 -4.076063e-02 -4.418380e-03  4.961894e-03  1.172514e-02
## [121]  1.279914e-02  1.559361e-02  9.903859e-03 -6.707119e-03 -3.481887e-02
## [126] -3.110829e-02 -3.903156e-02 -4.520254e-02  1.934928e-03 -1.476190e-02
## [131]  3.042967e-02 -4.119257e-04  9.975052e-03 -8.180574e-03 -1.426785e-02
## [136] -1.473459e-02 -1.262956e-02 -3.471981e-02  2.951784e-02  1.275773e-02
## [141]  2.605106e-02 -3.286098e-02  4.799486e-02  1.498859e-02  6.534830e-02
## [146]  8.494612e-03  3.379015e-02  1.136678e-02  9.227011e-03  3.926860e-02
```

```
## [151] -1.854547e-02  1.989188e-02  2.048464e-02 -5.257663e-03  1.693443e-02
## [156]  2.551796e-03  2.111729e-02 -2.633941e-02  5.215489e-02  9.304958e-03
## [161] -2.238706e-02  2.085424e-02  9.963315e-03 -3.918847e-02 -1.005742e-02
## [166]  5.642623e-03 -2.215043e-02  1.676637e-02 -4.108872e-02  1.085121e-02
## [171]  2.345948e-02 -1.101251e-02  3.893472e-02  3.347142e-03 -2.524497e-02
## [176] -1.534999e-02  2.713715e-02  9.312150e-04  3.543066e-02 -2.305974e-03
## [181] -8.470446e-03 -1.377566e-02 -1.856217e-02  1.570351e-02 -3.988699e-02
## [186] -7.239153e-03  4.828341e-03  2.529068e-02  2.448886e-03  3.970908e-04
## [191] -5.115926e-02  1.727974e-02  5.793265e-02  1.565454e-02 -2.390163e-02
## [196]  6.839563e-02 -2.387001e-02  1.083306e-02 -1.534612e-03  2.639429e-02
## [201]  1.371805e-02 -2.251611e-02 -1.115526e-02  3.583521e-03  1.423739e-02
## [206] -1.308146e-02  2.141635e-02 -3.788128e-02 -1.741954e-02 -1.711938e-03
## [211]  1.226354e-02 -1.228665e-02 -9.241076e-03 -2.519434e-02  1.316024e-02
## [216]  3.396107e-02  4.902241e-03 -3.204106e-03  1.725211e-02 -2.831802e-02
## [221]  3.617940e-02 -3.424370e-03 -1.971278e-02 -5.028879e-02 -1.186034e-03
## [226] -4.267172e-03 -8.851784e-03  6.371727e-04 -1.480511e-02  2.237151e-03
## [231]  1.059301e-02  1.529504e-02  2.798716e-02  2.113044e-03  4.627354e-02
## [236]  2.150548e-02  1.552263e-02 -2.243920e-02  1.403152e-03  5.985821e-02
## [241]  6.708601e-02 -5.931400e-03  2.322788e-02  1.822676e-02  5.470900e-03
## [246] -1.981885e-02 -1.887986e-02 -3.662484e-02 -1.400290e-02 -5.818905e-03
## [251] -7.868672e-03  4.870158e-02  3.235950e-02 -8.914801e-03  7.853349e-03
## [256] -1.651352e-02 -4.104373e-02  5.157024e-02  1.634313e-02  1.173142e-02
## [261]  1.888517e-02  4.517487e-03 -2.026799e-03  6.581309e-02 -1.553143e-02
## [266] -1.663876e-02  1.798302e-02  2.791401e-02  2.400681e-02  1.546458e-02
## [271] -3.640613e-02  1.807492e-03  6.218620e-03  1.459915e-02  5.346046e-02
## [276]  1.451935e-02  2.059760e-02 -1.278356e-02 -3.364456e-02  9.724260e-03
## [281]  1.119750e-03 -2.174072e-03  2.086653e-02  7.785686e-03 -2.562187e-02
## [286] -3.030858e-02  2.376484e-02  2.361198e-02 -2.567666e-02  1.264876e-02
## [291]  6.000163e-03 -1.380895e-03  3.462698e-03  1.741267e-02 -3.205232e-02
## [296] -3.066369e-02 -2.560239e-02 -3.886442e-02 -3.738674e-04 -2.667464e-02
## [301] -3.233380e-04 -1.905022e-03  8.119603e-03 -1.810339e-03  1.326173e-02
## [306]  8.784828e-03 -2.545738e-02  5.060447e-02  2.237830e-02 -3.618737e-02
## [311] -1.152640e-02  3.080213e-02 -1.443041e-02 -1.980447e-02 -2.501438e-02
## [316]  1.616711e-02 -2.494761e-02  1.167857e-02 -3.789242e-02 -1.462202e-02
## [321]  2.925621e-02  6.236068e-03  2.182136e-02  2.243749e-03  1.158926e-03
## [326] -1.364280e-03  4.607204e-02  1.929008e-02 -2.015348e-02  1.714925e-02
## [331]  2.677076e-02 -8.419070e-02 -9.418611e-03  1.176590e-03  5.396215e-03
## [336]  4.390033e-02  6.626598e-03  1.934385e-02 -2.408074e-03 -1.302730e-02
## [341] -2.471497e-03  2.147118e-02  5.675213e-03  2.211042e-02  1.639485e-03
## [346] -1.256361e-02  1.973128e-02 -2.072140e-02 -6.474758e-03 -1.455903e-02
## [351]  3.840702e-02  5.034090e-03  3.462368e-02  4.569145e-02  5.027182e-02
## [356] -2.565654e-02  5.011992e-02  1.043141e-03 -2.304482e-02  3.351899e-02
## [361]  1.398722e-02  1.480388e-02 -8.543116e-03  7.549366e-03 -3.389744e-02
## [366] -4.134542e-02  1.267130e-02 -6.667704e-03  4.408714e-02  1.730877e-02
## [371] -2.918504e-03  8.293627e-03 -5.458510e-03 -3.970267e-04 -2.876620e-02
## [376] -1.992555e-02  2.099078e-02 -1.929393e-02 -1.969610e-02 -1.809305e-02
## [381] -2.985892e-02  2.029462e-03  2.334025e-03  4.710495e-03 -1.148788e-02
## [386]  7.123171e-04  4.947452e-02 -2.647446e-02 -3.340622e-02 -1.251763e-02
## [391]  8.443487e-04  4.300186e-02  1.053637e-02 -3.673824e-02  1.699945e-02
## [396]  4.612986e-02  1.781799e-02 -7.433761e-03  8.579744e-03  3.066870e-02
## [401]  1.073631e-02 -2.622918e-02  5.136574e-04  5.274748e-02 -1.690665e-02
## [406] -4.388955e-02  4.628706e-03  1.158199e-02  3.447335e-02 -1.318912e-02
## [411] -5.825247e-03  1.061814e-02  2.220591e-02 -2.631666e-02  2.950213e-02
## [416]  1.959744e-02  1.714243e-03  1.039816e-02 -1.126247e-03 -1.919047e-02
```

```
## [421] -5.322368e-02 -2.481086e-02 -9.103783e-04 -2.931030e-02  6.968363e-05
## [426] -3.833379e-03 -6.850220e-03 -2.121231e-02 -2.885730e-02 -3.379744e-02
## [431]  4.884193e-03 -5.112231e-02 -3.679587e-03 -2.598087e-02  5.489897e-02
## [436] -2.531746e-02  2.477322e-03 -1.484469e-02 -2.125184e-03 -5.614878e-03
## [441] -1.437473e-02 -7.169370e-03  7.358067e-03  1.764329e-02  3.219762e-02
## [446] -4.753034e-03  1.928877e-02  4.850588e-03  1.345971e-02  2.697716e-02
## [451] -4.479156e-03  4.695062e-03 -8.329566e-03 -3.157995e-02  2.688582e-02
## [456] -3.213639e-02 -2.252116e-02 -3.753219e-02 -1.818339e-02 -5.605059e-03
## [461] -9.459691e-03  8.230368e-03 -2.557582e-02
```

```
h_1 <- hist(diff_AB, ylim = c(0,200))
text(h_1$mids,h_1$counts,labels=h_1$counts, adj=c(0.5, -0.5))
```




```
prob <- 236/463
prob
```

```
## [1] 0.5097192
```

The probability that A will have a higher evaluation is 0.51

10.8 How many simulation draws:

Take the model from Exercise 10.6 that predicts course evaluations from beauty and other predictors.

10.8a

Display and discuss the fitted model. Focus on the estimate and standard error for the coefficient of beauty. use the second model as well

```r
fit_106b_2 <- stan_glm(y_106a ~ x_106a + x_age + g_106b)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.

##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.1e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.21 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.076167 seconds (Warm-up)
## Chain 1:                0.096582 seconds (Sampling)
## Chain 1:                0.172749 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 5.6e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.56 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.062106 seconds (Warm-up)
```

```
## Chain 2:                0.089406 seconds (Sampling)
## Chain 2:                0.151512 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.29 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 3: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 3: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 3: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 3: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 3: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 3: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 3: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 3: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 3: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 3: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 3: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.059028 seconds (Warm-up)
## Chain 3:                0.093412 seconds (Sampling)
## Chain 3:                0.15244 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.4e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.14 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 4: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 4: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 4: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 4: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 4: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 4: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 4: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
## Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
## Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.056775 seconds (Warm-up)
## Chain 4:                0.115305 seconds (Sampling)
## Chain 4:                0.17208 seconds (Total)
## Chain 4:
```

```r
fit_106b_2
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b
##  observations: 463
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept)  4.2    0.1
## x_106a       0.1    0.0
## x_age        0.0    0.0
## g_106b      -0.2    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.5    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

10.8b

Compute the median and mad sd of the posterior simulations of the coefficient of beauty, and check that these are the same as the output from printing the fit.

```r
M1 <- fit_106b_2
sims <- as.matrix(M1)
Median <- apply(sims, 2, median)
MAD_SD <- apply(sims, 2, mad)
print(M1)
```


```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b
##  observations: 463
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept)  4.2    0.1
## x_106a       0.1    0.0
## x_age        0.0    0.0
## g_106b      -0.2    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.5    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```
Median
```

```
##   (Intercept)       x_106a        x_age       g_106b        sigma 
##   4.218235972  0.140764352 -0.002458925 -0.208563323  0.537671997
```

```
MAD_SD
```

```
## (Intercept)      x_106a       x_age      g_106b       sigma 
## 0.143771963 0.032704798 0.002819363 0.053171762 0.017703229
```

10.8c

Fit again, this time setting iter = 1000 in your stan_glm call. Do this a few times in order to get a sense of the simulation variability.

```
fit_108c <- stan_glm(y_106a ~ x_106a+x_age+g_106b, iter = 1000)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
## 
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1: 
## Chain 1: Gradient evaluation took 2.5e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.25 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1: 
## Chain 1: 
## Chain 1: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 1: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 1: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 1: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 1: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 1: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 1: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 1: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 1: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 1: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 1: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 1: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 1: 
## Chain 1:  Elapsed Time: 0.036637 seconds (Warm-up)
## Chain 1:                0.057298 seconds (Sampling)
## Chain 1:                0.093935 seconds (Total)
## Chain 1: 
## 
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2: 
## Chain 2: Gradient evaluation took 1.3e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.13 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2: 
## Chain 2: 
```

```
## Chain 2: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 2: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 2: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 2: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 2: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 2: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 2: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 2: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 2: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 2: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 2: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 2: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.034661 seconds (Warm-up)
## Chain 2:                0.041769 seconds (Sampling)
## Chain 2:                0.07643 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.2e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 3: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 3: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 3: Iteration: 300 / 1000 [ 30%]  (Warmup)
## Chain 3: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 3: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 3: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 3: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 3: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 3: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 3: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 3: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.030762 seconds (Warm-up)
## Chain 3:                0.042187 seconds (Sampling)
## Chain 3:                0.072949 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.6e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: Iteration:   1 / 1000 [  0%]  (Warmup)
## Chain 4: Iteration: 100 / 1000 [ 10%]  (Warmup)
## Chain 4: Iteration: 200 / 1000 [ 20%]  (Warmup)
## Chain 4: Iteration: 300 / 1000 [ 30%]  (Warmup)
```

```
## Chain 4: Iteration: 400 / 1000 [ 40%]  (Warmup)
## Chain 4: Iteration: 500 / 1000 [ 50%]  (Warmup)
## Chain 4: Iteration: 501 / 1000 [ 50%]  (Sampling)
## Chain 4: Iteration: 600 / 1000 [ 60%]  (Sampling)
## Chain 4: Iteration: 700 / 1000 [ 70%]  (Sampling)
## Chain 4: Iteration: 800 / 1000 [ 80%]  (Sampling)
## Chain 4: Iteration: 900 / 1000 [ 90%]  (Sampling)
## Chain 4: Iteration: 1000 / 1000 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.031155 seconds (Warm-up)
## Chain 4:                0.041501 seconds (Sampling)
## Chain 4:                0.072656 seconds (Total)
## Chain 4:
```

```
print(fit_108c)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b
##  observations: 463
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept)  4.2    0.2
## x_106a       0.1    0.0
## x_age        0.0    0.0
## g_106b      -0.2    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.5    0.0
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

10.8d

Repeat the previous step, setting iter = 100 and then iter = 10.

```
fit_108d_1 <- stan_glm(y_106a ~ x_106a+x_age+g_106b, iter = 100)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 3.9e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.39 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
```

```
## Chain 1: WARNING: There aren't enough warmup iterations to fit the
## Chain 1:          three stages of adaptation as currently configured.
## Chain 1:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 1:          the given number of warmup iterations:
## Chain 1:            init_buffer = 7
## Chain 1:            adapt_window = 38
## Chain 1:            term_buffer = 5
## Chain 1:
## Chain 1: Iteration:  1 / 100 [  1%]  (Warmup)
## Chain 1: Iteration: 10 / 100 [ 10%]  (Warmup)
## Chain 1: Iteration: 20 / 100 [ 20%]  (Warmup)
## Chain 1: Iteration: 30 / 100 [ 30%]  (Warmup)
## Chain 1: Iteration: 40 / 100 [ 40%]  (Warmup)
## Chain 1: Iteration: 50 / 100 [ 50%]  (Warmup)
## Chain 1: Iteration: 51 / 100 [ 51%]  (Sampling)
## Chain 1: Iteration: 60 / 100 [ 60%]  (Sampling)
## Chain 1: Iteration: 70 / 100 [ 70%]  (Sampling)
## Chain 1: Iteration: 80 / 100 [ 80%]  (Sampling)
## Chain 1: Iteration: 90 / 100 [ 90%]  (Sampling)
## Chain 1: Iteration: 100 / 100 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.005884 seconds (Warm-up)
## Chain 1:                0.005148 seconds (Sampling)
## Chain 1:                0.011032 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1.6e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: WARNING: There aren't enough warmup iterations to fit the
## Chain 2:          three stages of adaptation as currently configured.
## Chain 2:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 2:          the given number of warmup iterations:
## Chain 2:            init_buffer = 7
## Chain 2:            adapt_window = 38
## Chain 2:            term_buffer = 5
## Chain 2:
## Chain 2: Iteration:  1 / 100 [  1%]  (Warmup)
## Chain 2: Iteration: 10 / 100 [ 10%]  (Warmup)
## Chain 2: Iteration: 20 / 100 [ 20%]  (Warmup)
## Chain 2: Iteration: 30 / 100 [ 30%]  (Warmup)
## Chain 2: Iteration: 40 / 100 [ 40%]  (Warmup)
## Chain 2: Iteration: 50 / 100 [ 50%]  (Warmup)
## Chain 2: Iteration: 51 / 100 [ 51%]  (Sampling)
## Chain 2: Iteration: 60 / 100 [ 60%]  (Sampling)
## Chain 2: Iteration: 70 / 100 [ 70%]  (Sampling)
## Chain 2: Iteration: 80 / 100 [ 80%]  (Sampling)
## Chain 2: Iteration: 90 / 100 [ 90%]  (Sampling)
## Chain 2: Iteration: 100 / 100 [100%]  (Sampling)
## Chain 2:
```

```
## Chain 2:  Elapsed Time: 0.002271 seconds (Warm-up)
## Chain 2:                0.004538 seconds (Sampling)
## Chain 2:                0.006809 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: WARNING: There aren't enough warmup iterations to fit the
## Chain 3:          three stages of adaptation as currently configured.
## Chain 3:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 3:          the given number of warmup iterations:
## Chain 3:            init_buffer = 7
## Chain 3:            adapt_window = 38
## Chain 3:            term_buffer = 5
## Chain 3:
## Chain 3: Iteration:  1 / 100 [  1%]  (Warmup)
## Chain 3: Iteration: 10 / 100 [ 10%]  (Warmup)
## Chain 3: Iteration: 20 / 100 [ 20%]  (Warmup)
## Chain 3: Iteration: 30 / 100 [ 30%]  (Warmup)
## Chain 3: Iteration: 40 / 100 [ 40%]  (Warmup)
## Chain 3: Iteration: 50 / 100 [ 50%]  (Warmup)
## Chain 3: Iteration: 51 / 100 [ 51%]  (Sampling)
## Chain 3: Iteration: 60 / 100 [ 60%]  (Sampling)
## Chain 3: Iteration: 70 / 100 [ 70%]  (Sampling)
## Chain 3: Iteration: 80 / 100 [ 80%]  (Sampling)
## Chain 3: Iteration: 90 / 100 [ 90%]  (Sampling)
## Chain 3: Iteration: 100 / 100 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.002354 seconds (Warm-up)
## Chain 3:                0.005287 seconds (Sampling)
## Chain 3:                0.007641 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.5e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.15 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: WARNING: There aren't enough warmup iterations to fit the
## Chain 4:          three stages of adaptation as currently configured.
## Chain 4:          Reducing each adaptation stage to 15%/75%/10% of
## Chain 4:          the given number of warmup iterations:
## Chain 4:            init_buffer = 7
## Chain 4:            adapt_window = 38
## Chain 4:            term_buffer = 5
## Chain 4:
## Chain 4: Iteration:  1 / 100 [  1%]  (Warmup)
```

```
## Chain 4: Iteration: 10 / 100 [ 10%]  (Warmup)
## Chain 4: Iteration: 20 / 100 [ 20%]  (Warmup)
## Chain 4: Iteration: 30 / 100 [ 30%]  (Warmup)
## Chain 4: Iteration: 40 / 100 [ 40%]  (Warmup)
## Chain 4: Iteration: 50 / 100 [ 50%]  (Warmup)
## Chain 4: Iteration: 51 / 100 [ 51%]  (Sampling)
## Chain 4: Iteration: 60 / 100 [ 60%]  (Sampling)
## Chain 4: Iteration: 70 / 100 [ 70%]  (Sampling)
## Chain 4: Iteration: 80 / 100 [ 80%]  (Sampling)
## Chain 4: Iteration: 90 / 100 [ 90%]  (Sampling)
## Chain 4: Iteration: 100 / 100 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.003823 seconds (Warm-up)
## Chain 4:                0.008543 seconds (Sampling)
## Chain 4:                0.012366 seconds (Total)
## Chain 4:

## Warning: Bulk Effective Samples Size (ESS) is too low, indicating posterior means and medians may be
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#bulk-ess

## Warning: Tail Effective Samples Size (ESS) is too low, indicating posterior variances and tail quant
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#tail-ess
```

```
fit_108d_2 <- stan_glm(y_106a ~ x_106a+x_age+g_106b, iter = 10)
```

```
## Warning: Omitting the 'data' argument is not recommended and may not be
## allowed in future versions of rstanarm. Some post-estimation functions (in
## particular 'update', 'loo', 'kfold') are not guaranteed to work properly
## unless 'data' is specified as a data frame.

##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.9e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: WARNING: No variance estimation is
## Chain 1:          performed for num_warmup < 20
## Chain 1:
## Chain 1: Iteration: 1 / 10 [ 10%]  (Warmup)
## Chain 1: Iteration: 2 / 10 [ 20%]  (Warmup)
## Chain 1: Iteration: 3 / 10 [ 30%]  (Warmup)
## Chain 1: Iteration: 4 / 10 [ 40%]  (Warmup)
## Chain 1: Iteration: 5 / 10 [ 50%]  (Warmup)
## Chain 1: Iteration: 6 / 10 [ 60%]  (Sampling)
## Chain 1: Iteration: 7 / 10 [ 70%]  (Sampling)
## Chain 1: Iteration: 8 / 10 [ 80%]  (Sampling)
## Chain 1: Iteration: 9 / 10 [ 90%]  (Sampling)
## Chain 1: Iteration: 10 / 10 [100%]  (Sampling)
## Chain 1:
## Chain 1:  Elapsed Time: 0.000285 seconds (Warm-up)
## Chain 1:                0.000283 seconds (Sampling)
```

```
## Chain 1:                0.000568 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.2e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.22 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: WARNING: No variance estimation is
## Chain 2:          performed for num_warmup < 20
## Chain 2:
## Chain 2: Iteration: 1 / 10 [ 10%]  (Warmup)
## Chain 2: Iteration: 2 / 10 [ 20%]  (Warmup)
## Chain 2: Iteration: 3 / 10 [ 30%]  (Warmup)
## Chain 2: Iteration: 4 / 10 [ 40%]  (Warmup)
## Chain 2: Iteration: 5 / 10 [ 50%]  (Warmup)
## Chain 2: Iteration: 6 / 10 [ 60%]  (Sampling)
## Chain 2: Iteration: 7 / 10 [ 70%]  (Sampling)
## Chain 2: Iteration: 8 / 10 [ 80%]  (Sampling)
## Chain 2: Iteration: 9 / 10 [ 90%]  (Sampling)
## Chain 2: Iteration: 10 / 10 [100%]  (Sampling)
## Chain 2:
## Chain 2:  Elapsed Time: 0.000153 seconds (Warm-up)
## Chain 2:                0.000439 seconds (Sampling)
## Chain 2:                0.000592 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.6e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.16 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: WARNING: No variance estimation is
## Chain 3:          performed for num_warmup < 20
## Chain 3:
## Chain 3: Iteration: 1 / 10 [ 10%]  (Warmup)
## Chain 3: Iteration: 2 / 10 [ 20%]  (Warmup)
## Chain 3: Iteration: 3 / 10 [ 30%]  (Warmup)
## Chain 3: Iteration: 4 / 10 [ 40%]  (Warmup)
## Chain 3: Iteration: 5 / 10 [ 50%]  (Warmup)
## Chain 3: Iteration: 6 / 10 [ 60%]  (Sampling)
## Chain 3: Iteration: 7 / 10 [ 70%]  (Sampling)
## Chain 3: Iteration: 8 / 10 [ 80%]  (Sampling)
## Chain 3: Iteration: 9 / 10 [ 90%]  (Sampling)
## Chain 3: Iteration: 10 / 10 [100%]  (Sampling)
## Chain 3:
## Chain 3:  Elapsed Time: 0.000148 seconds (Warm-up)
## Chain 3:                0.000302 seconds (Sampling)
## Chain 3:                0.00045 seconds (Total)
## Chain 3:
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 1.2e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.12 seconds.
## Chain 4: Adjust your expectations accordingly!
## Chain 4:
## Chain 4:
## Chain 4: WARNING: No variance estimation is
## Chain 4:          performed for num_warmup < 20
## Chain 4:
## Chain 4: Iteration: 1 / 10 [ 10%]  (Warmup)
## Chain 4: Iteration: 2 / 10 [ 20%]  (Warmup)
## Chain 4: Iteration: 3 / 10 [ 30%]  (Warmup)
## Chain 4: Iteration: 4 / 10 [ 40%]  (Warmup)
## Chain 4: Iteration: 5 / 10 [ 50%]  (Warmup)
## Chain 4: Iteration: 6 / 10 [ 60%]  (Sampling)
## Chain 4: Iteration: 7 / 10 [ 70%]  (Sampling)
## Chain 4: Iteration: 8 / 10 [ 80%]  (Sampling)
## Chain 4: Iteration: 9 / 10 [ 90%]  (Sampling)
## Chain 4: Iteration: 10 / 10 [100%]  (Sampling)
## Chain 4:
## Chain 4:  Elapsed Time: 0.00011 seconds (Warm-up)
## Chain 4:                0.000296 seconds (Sampling)
## Chain 4:                0.000406 seconds (Total)
## Chain 4:

## Warning: There were 15 divergent transitions after warmup. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.

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

## Warning: The largest R-hat is Inf, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## http://mc-stan.org/misc/warnings.html#r-hat

## Warning: Markov chains did not converge! Do not analyze results!
```

```r
print(fit_108d_1)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b
##  observations: 463
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept)  4.2    0.1
## x_106a       0.1    0.0
## x_age        0.0    0.0
## g_106b      -0.2    0.1
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.5    0.0
##
```

```
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

```r
print(fit_108d_2)
```

```
## stan_glm
##  family:       gaussian [identity]
##  formula:      y_106a ~ x_106a + x_age + g_106b
##  observations: 463
##  predictors:   4
## ------
##             Median MAD_SD
## (Intercept)  3.1    6.0
## x_106a       0.3    1.4
## x_age       -0.1    0.1
## g_106b       2.5    1.7
##
## Auxiliary parameter(s):
##       Median MAD_SD
## sigma 0.3    0.3
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

10.8e

How many simulations were needed to give a good approximation to the mean and standard error for the coefficient of beauty?

At least 100 simulations