HW 1 Solutions

Chang Lu

Fall 2024

7.2 Fake-data simulation and regression:

Simulate 100 data points from the linear model, $y = a + bx + \text{error}$, 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.

```r
set.seed(1)

n <- 100
a <- 5
b <- 7

x <- runif(n, min =0, max = 50)
error <- rnorm(n, mean = 0, sd =3)

y <- a+b*x +error

model <- lm(y ~ x)

summary(model)
```

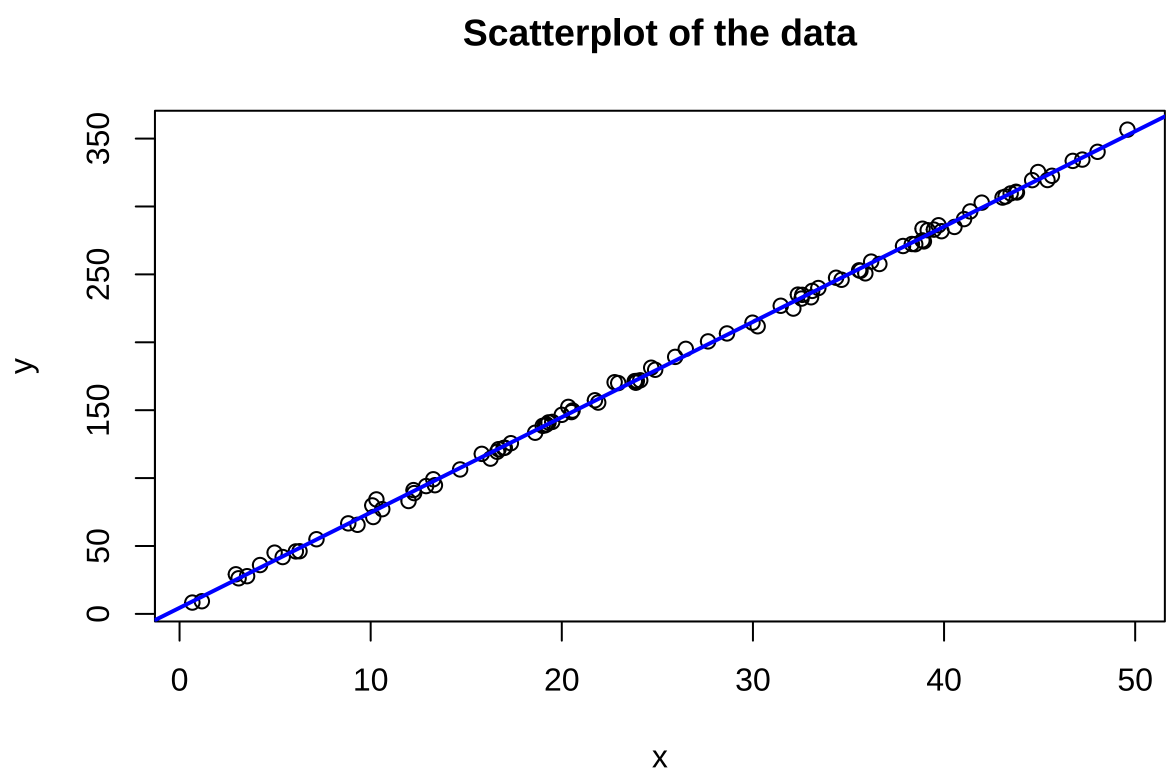
```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.5493 -1.6867 -0.2612  1.5728  7.5498
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.46202    0.61744   7.227 1.09e-10 ***
## x            7.01874    0.02121 330.956  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 2.823 on 98 degrees of freedom
## Multiple R-squared:  0.9991, Adjusted R-squared:  0.9991
## F-statistic: 1.095e+05 on 1 and 98 DF,  p-value: < 2.2e-16
```

7.2b

Graph a scatterplot of the data and the regression line.

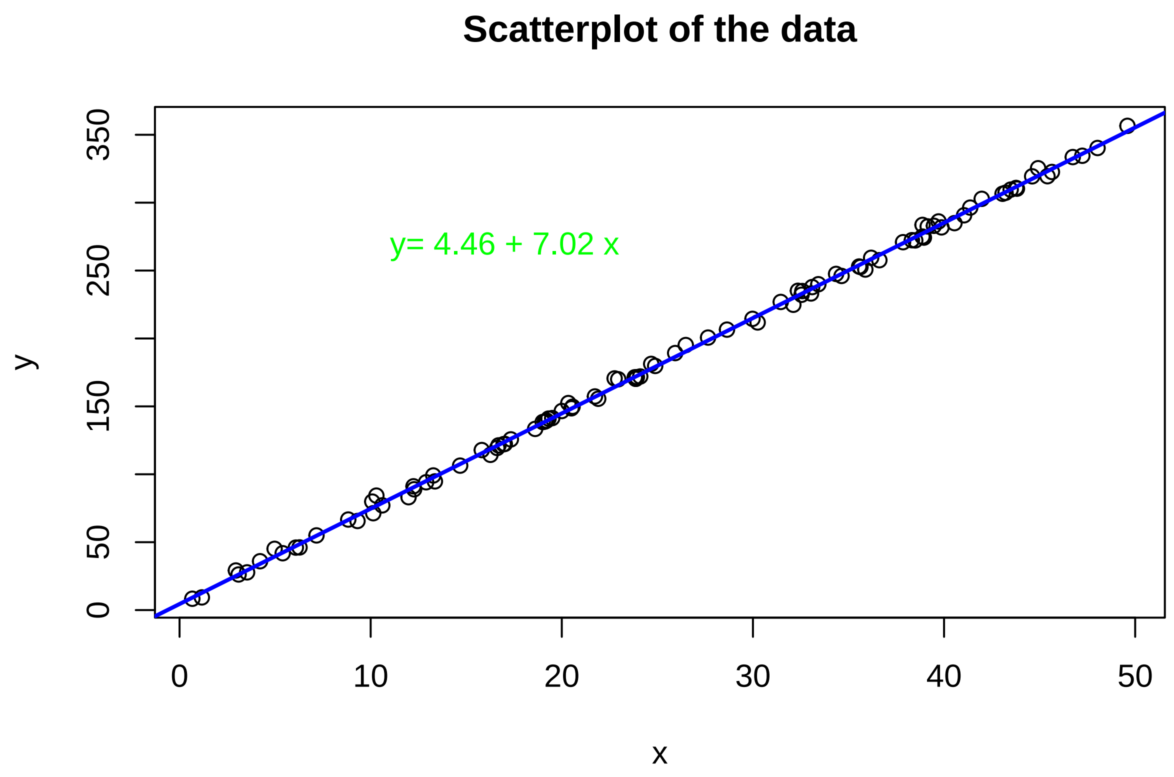
```
plot(x, y, main = "Scatterplot of the data", xlab = "x", ylab = "y")

abline(model, col ="blue", lwd=2)
```



7.2c

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

```
plot(x, y, main = "Scatterplot of the data", xlab = "x", ylab = "y")

abline(model, col ="blue", lwd=2)
coefficients <- coef(model)

intercept <- round(coefficients[1], 2)
slope <- round(coefficients[2], 2)

text(x=10, y =3/4*max(y), labels = paste("y=", intercept, "+", slope, "x" ), col ="green", pos=4)
```



7.3 Fake-data simulation and fitting the wrong model:

Simulate 100 data points from the model $y = a + bx + cx^2$ + 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 the output.

```
set.seed(1)

n <- 100
a <- 3
b <- 5
c <- 7

x1 <- runif(n, min=0, max = 50)
error1 <- rnorm(n, mean = 0, sd =3)

y1 <- a+b*x1 + c*x1^2+error1

linear_model <- stan_glm(y1 ~ x1)
```

```
## 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 5.3e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.53 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: 0.599 seconds (Warm-up)
## Chain 1:                0.036 seconds (Sampling)
## Chain 1:                0.635 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 9e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.68 seconds (Warm-up)
## Chain 2:                0.036 seconds (Sampling)
## Chain 2:                0.716 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 9e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.508 seconds (Warm-up)
## Chain 3:                0.034 seconds (Sampling)
## Chain 3:                0.542 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 9e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.317 seconds (Warm-up)
## Chain 4:                0.036 seconds (Sampling)
## Chain 4:                0.353 seconds (Total)
## Chain 4:
```

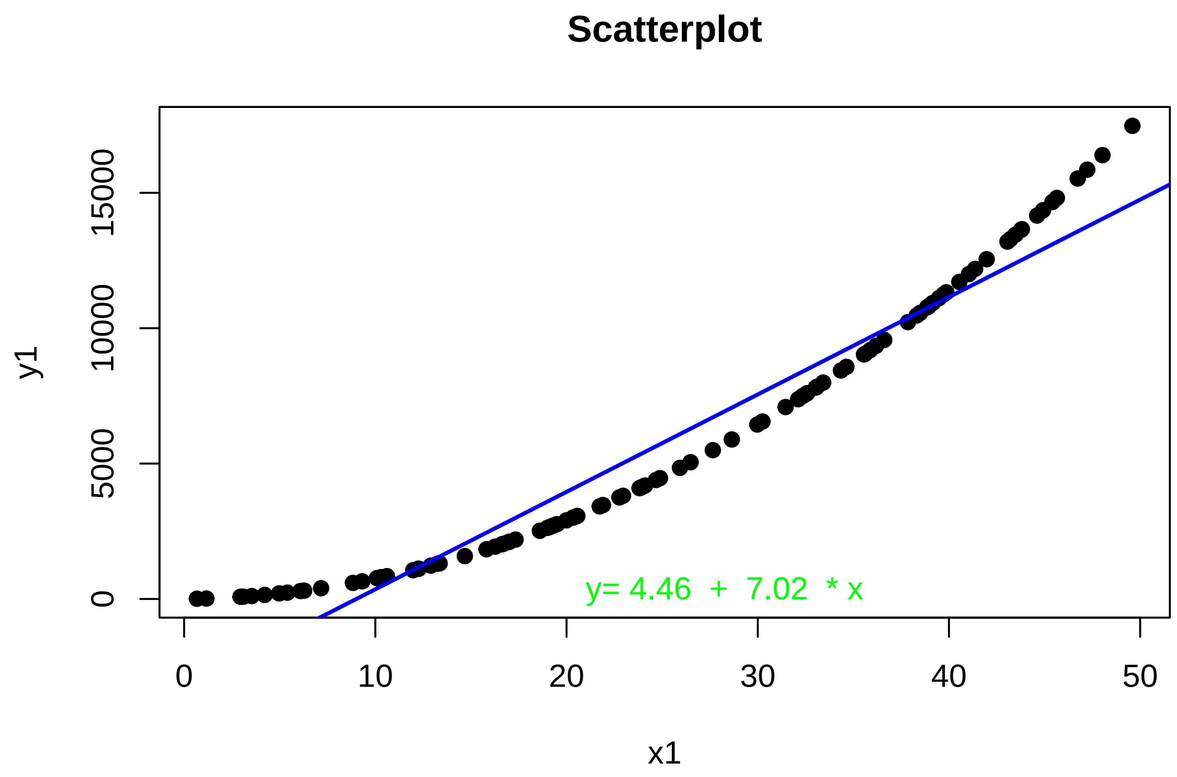
```r
summary(linear_model)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y1 ~ x1
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 100
##  predictors:   2
##
## Estimates:
##               mean    sd      10%     50%     90%
## (Intercept) -3242.4   265.7 -3578.3 -3241.4 -2905.3
## x1            359.6     9.2   347.7   359.8   371.2
## sigma        1170.6    86.6  1061.7  1166.6  1284.7
##
## Fit Diagnostics:
##            mean   sd     10%    50%    90%
## mean_PPD 6068.7  163.9 5857.4 6067.2 6278.0
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   4.3  1.0  3743
## x1            0.1  1.0  3830
## sigma         1.4  1.0  3723
## mean_PPD      2.9  1.0  3206
## log-posterior 0.0  1.0  1578
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

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?

```r
plot(x1, y1, main = "Scatterplot", xlab = "x1", ylab = "y1", pch=19)

abline(linear_model, col="blue", lwd=2)

coefficients1 <- coef(linear_model)

intercept1 <- round(coefficients1[1], 2)
slope1 <- round(coefficients1[2], 2)
text(x=20, y=3/4*max(y), labels=paste("y=", intercept, " + ", slope, " * x"), col="green", pos =4)
```



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%.

```r
hibbs <- read.table("hibbs.dat", header = TRUE)

hibbs
```

```
##    year growth  vote inc_party_candidate other_candidate
## 1  1952   2.40 44.60          Stevenson      Eisenhower
## 2  1956   2.89 57.76         Eisenhower       Stevenson
## 3  1960   0.85 49.91              Nixon         Kennedy
## 4  1964   4.21 61.34            Johnson       Goldwater
## 5  1968   3.02 49.60           Humphrey           Nixon
## 6  1972   3.62 61.79              Nixon        McGovern
## 7  1976   1.08 48.95               Ford          Carter
## 8  1980  -0.39 44.70             Carter          Reagan
## 9  1984   3.86 59.17             Reagan         Mondale
## 10 1988   2.27 53.94         Bush, Sr.        Dukakis
## 11 1992   0.38 46.55         Bush, Sr.        Clinton
## 12 1996   1.04 54.74            Clinton            Dole
## 13 2000   2.36 50.27               Gore      Bush, Jr.
## 14 2004   1.72 51.24         Bush, Jr.           Kerry
## 15 2008   0.10 46.32             McCain           Obama
## 16 2012   0.95 52.00               Obama          Romney
```

```
head(hibbs)
```

```
##   year growth  vote inc_party_candidate other_candidate
## 1 1952   2.40 44.60           Stevenson      Eisenhower
## 2 1956   2.89 57.76          Eisenhower       Stevenson
## 3 1960   0.85 49.91               Nixon         Kennedy
## 4 1964   4.21 61.34             Johnson       Goldwater
## 5 1968   3.02 49.60            Humphrey           Nixon
## 6 1972   3.62 61.79               Nixon        McGovern
```

```
hibbs$income_binary <- ifelse(hibbs$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.

```
mean_group_0 <- mean(hibbs$vote[hibbs$income_binary == 0])
mean_group_1 <- mean(hibbs$vote[hibbs$income_binary == 1])
mean_diff <- mean_group_1 - mean_group_0
se_group_0 <- sd(hibbs$vote[hibbs$income_binary == 0]) / sqrt(sum(hibbs$income_binary == 0))
se_group_1 <- sd(hibbs$vote[hibbs$income_binary == 1]) / sqrt(sum(hibbs$income_binary == 1))
se_diff <- sqrt(se_group_0^2 + se_group_1^2)

mean_diff
```

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

```
se_diff
```

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

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.

```
model <- lm(vote ~ income_binary, data = hibbs)

summary(model)
```

```
##
## Call:
## lm(formula = vote ~ income_binary, data = hibbs)
##
## 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 ***
## income_binary    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
```

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.

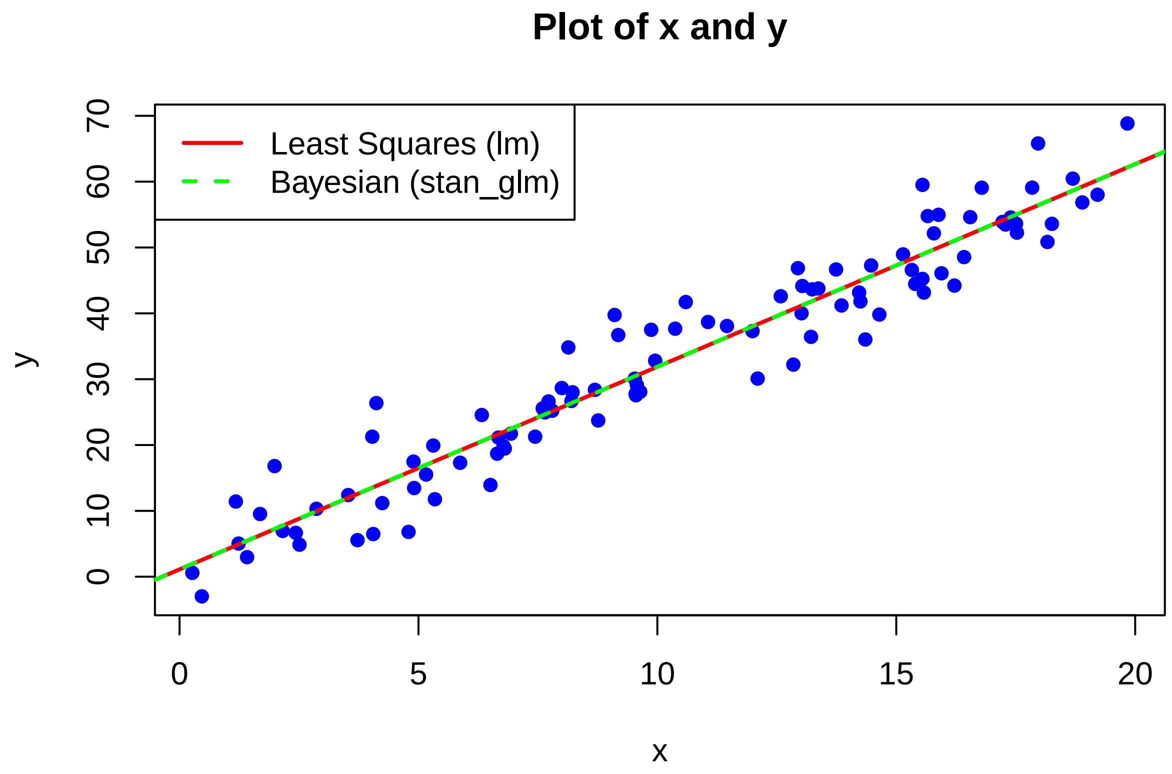
```r
library(rstanarm)

set.seed(1)

n <- 100
x <- runif(n, min = 0, max = 20)
error <- rnorm(n, mean = 0, sd = 5)

y <- 2 + 3 * x + error

lm_model <- lm(y ~ x)

stan_model <- stan_glm(y ~ x, family = gaussian, refresh = 0)
```

```
## 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.
```

8.8b

Plot the simulated data and the two fitted regression lines.

```r
plot(x, y, main = "Plot of x and y", xlab = "x", ylab = "y", pch = 16, col = "blue")


abline(lm_model, col = "red", lwd = 2, lty = 1)
abline(coef(stan_model)[1], coef(stan_model)[2], col = "green", lwd = 2, lty = 2)


legend("topleft", legend = c("Least Squares (lm)", "Bayesian (stan_glm)"), col = c("red", "green"), lwd
```



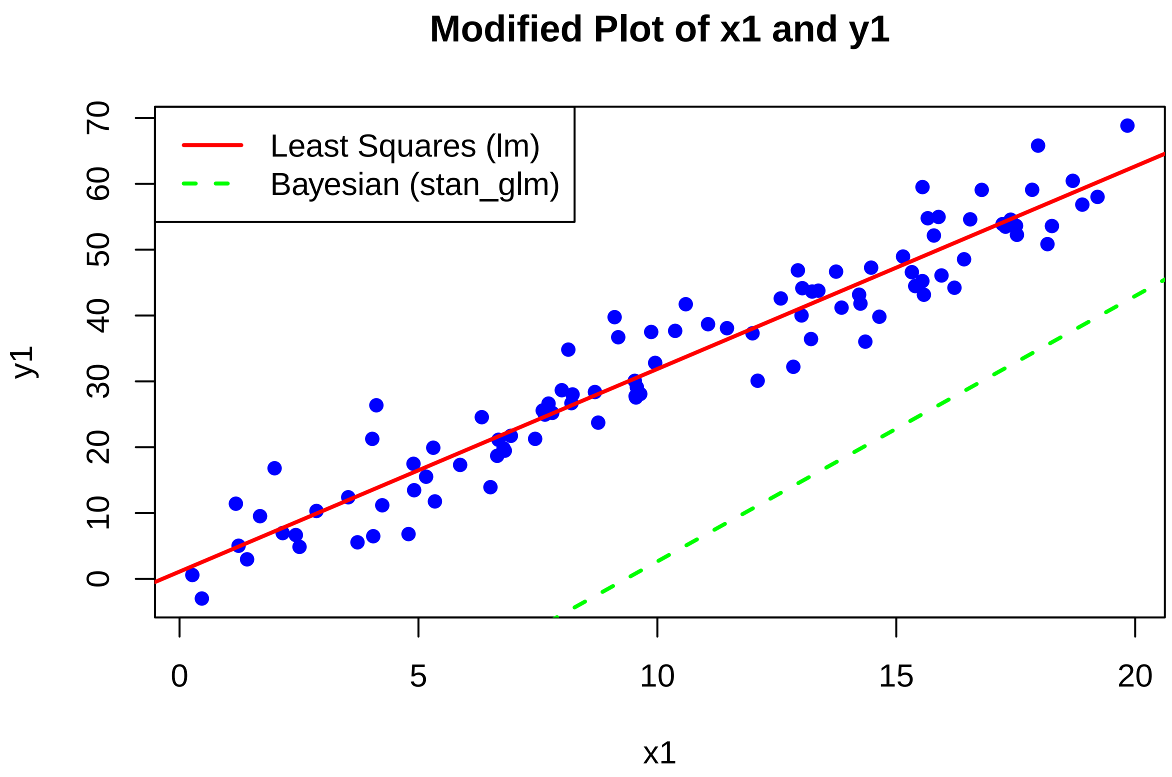
8.8c

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

```r
set.seed(1)

n1 <- 100
x1 <- runif(n1, min = 0, max = 20)
error1 <- rnorm(n1, mean = 0, sd = 5)

y1 <- 2 + 3 * x1 + error1

stan_model <- stan_glm(y1 ~ x1, family = gaussian,
                       prior_intercept = normal(0, 1),
                       prior = normal(5, 0.5),
                       prior_aux = exponential(1),
                       refresh = 0)
```

```
## 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.
```

```r
lm_model <- lm(y1 ~ x1)


plot(x, y, main = "Modified Plot of x1 and y1", xlab = "x1", ylab = "y1", pch = 16, col = "blue")


abline(lm_model, col = "red", lwd = 2, lty = 1)

abline(coef(stan_model)[1], coef(stan_model)[2], col = "green", lwd = 2, lty = 2)

legend("topleft", legend = c("Least Squares (lm)", "Bayesian (stan_glm)"), col = c("red", "green"), lwd
```



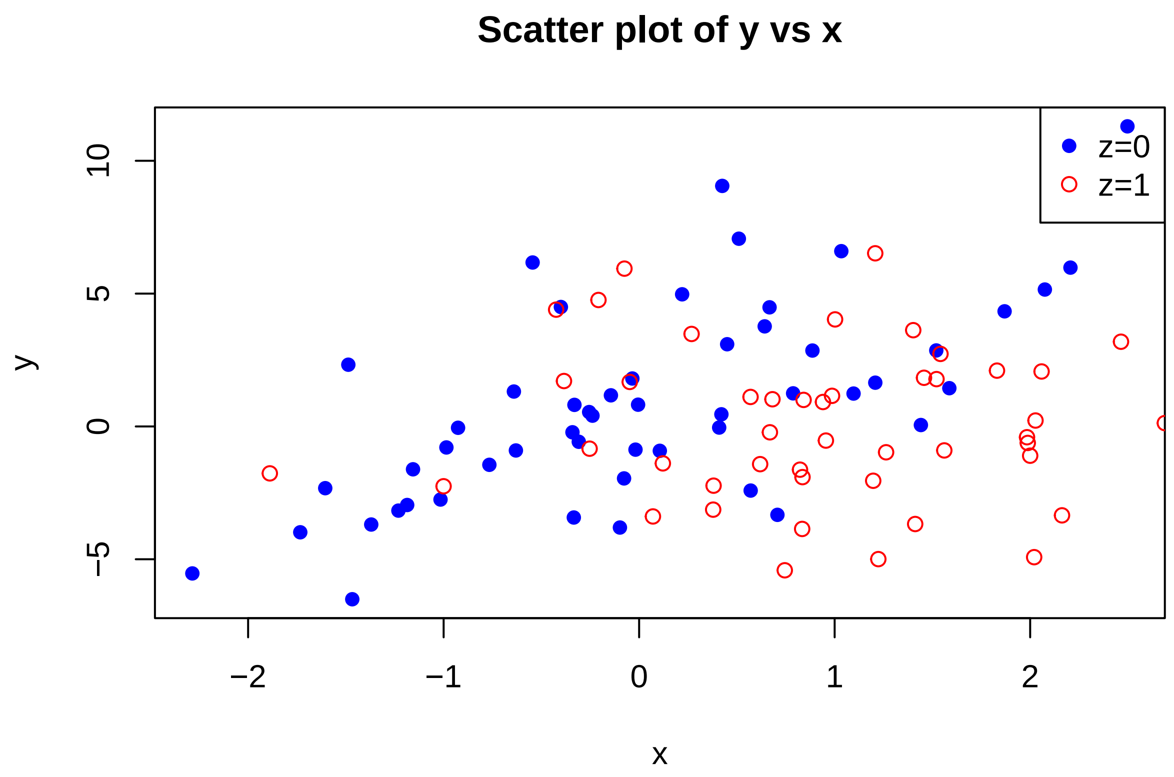
10.1 Regression with interactions:

Simulate 100 data points from the model, $y = b_0 + b_1x + b_2z + b_3xz$ + 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 z_i, equally likely to take on the values 0 and 1. Then draw x_i from a normal distribution with mean z_i and standard deviation 1. Then draw the error from its normal distribution and compute y_i.

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.

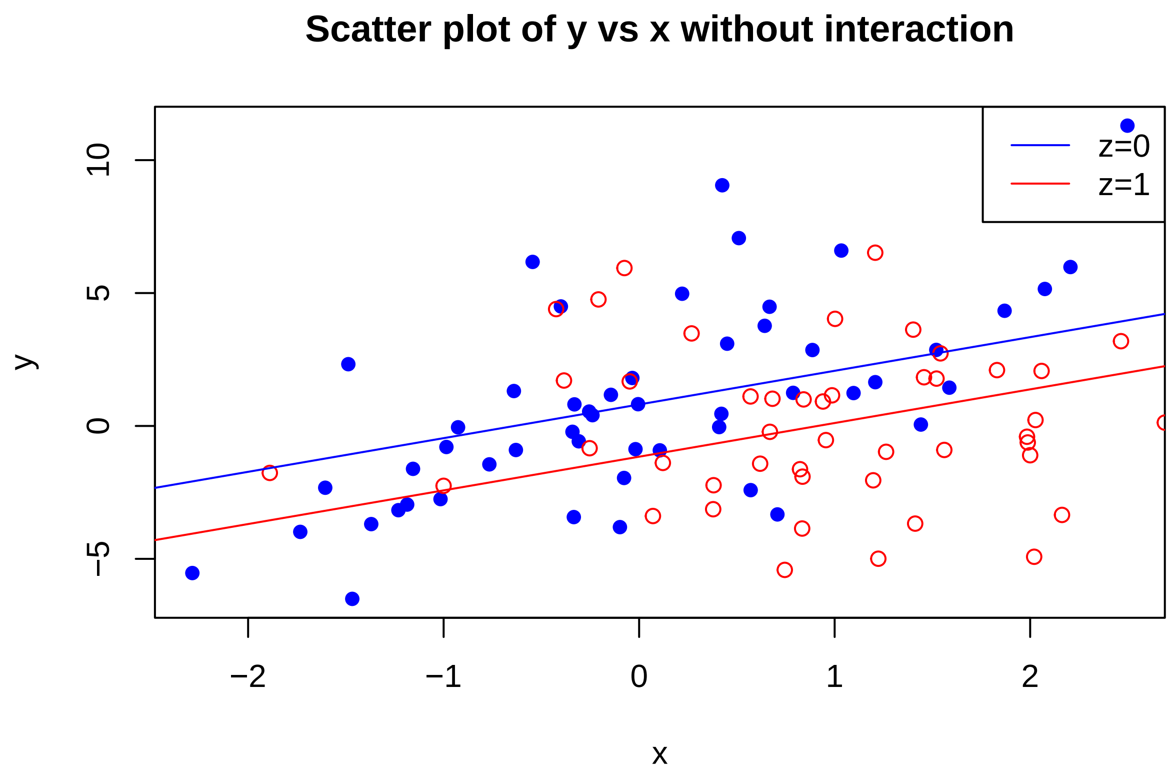
```r
set.seed(1)

b0 <- 1
b1 <- 2
b2 <- -1
b3 <- -2
n <- 100

z <- rbinom(n, 1, 0.5)
error <- rnorm(n, mean = 0, sd = 3)
x <- rnorm(n,mean = z, sd = 1)

y <- b0 + b1 * x + b2 * z + b3 * x * z + error

data <- data.frame(x = x, y = y, z = z)


plot(data$x[data$z == 0], data$y[data$z == 0], col = "blue", pch = 16, xlab = "x", ylab = "y", main = "S
points(data$x[data$z == 1], data$y[data$z == 1], col = "red", pch = 1)
legend("topright", legend = c("z=0", "z=1"), col = c("blue", "red"), pch = c(16, 1))
```



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.

```
model_nointeraction <- lm(y ~ x+z, data = data)

plot(data$x[data$z==0], data$y[data$z==0], col="blue", pch =16, xlab = "x", ylab = "y", main = "Scatter
points(data$x[data$z==1], data$y[data$z==1], col= "red", pch =1)

abline(coef(model_nointeraction)[1]+coef(model_nointeraction)[3]*0, coef(model_nointeraction)[2], col="
abline(coef(model_nointeraction)[1]+coef(model_nointeraction)[3]*1, coef(model_nointeraction)[2], col="

legend("topright", legend = c("z=0", "z=1"), col = c("blue", "red"), lty = 1)
```

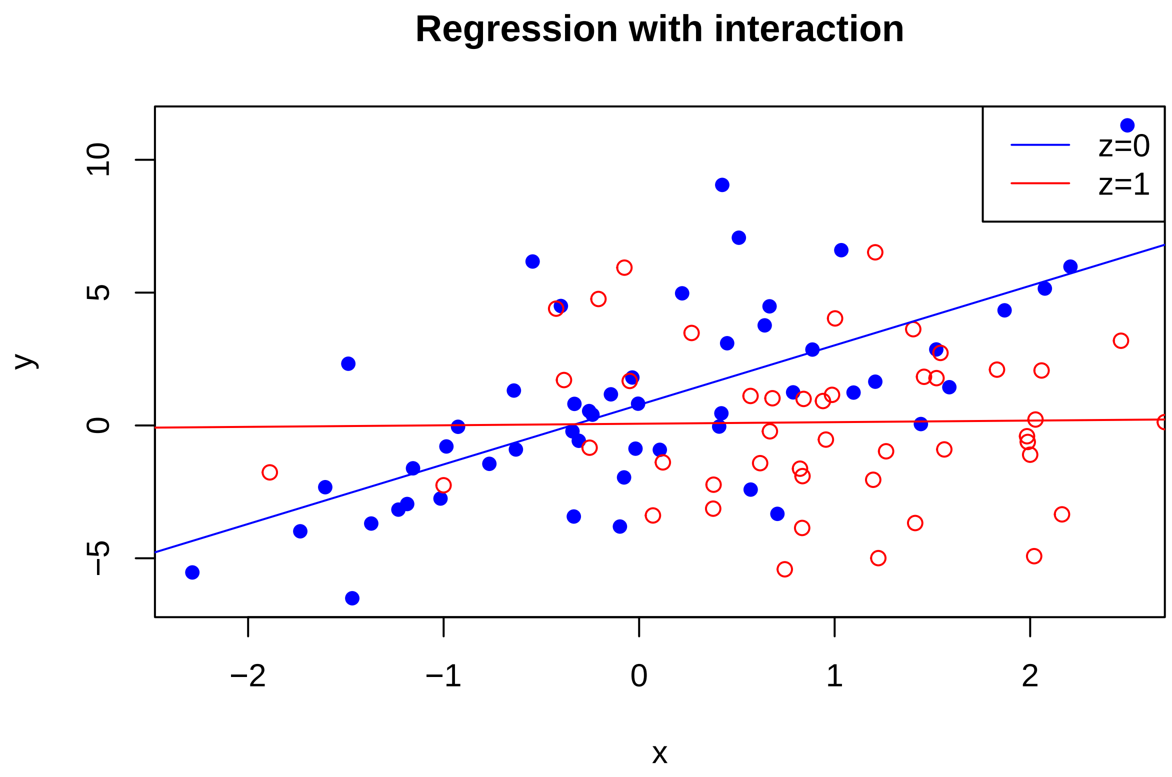


10.1c

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

```
model_interaction <- lm(y ~ x * z, data = data)

plot(data$x[data$z == 0], data$y[data$z == 0], col = "blue", pch = 16, xlab = "x", ylab = "y", main = "
points(data$x[data$z == 1], data$y[data$z == 1], col = "red", pch = 1)
```

```r
abline(coef(model_interaction)[1] + coef(model_interaction)[3] * 0, coef(model_interaction)[2] + coef(mo
abline(coef(model_interaction)[1] + coef(model_interaction)[3] * 1, coef(model_interaction)[2] + coef(mo

legend("topright", legend = c("z=0", "z=1"), col = c("blue", "red"), lty = 1)
```



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:

```
            Mediam MAD_SD
(Intercept) 1.2    0.2
x           1.6    0.4
z           2.7    0.3
x:z         0.7    0.5

Auxiliary parameter(s):
      Median MAD_SD
sigma 0.4    0.0
```

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.

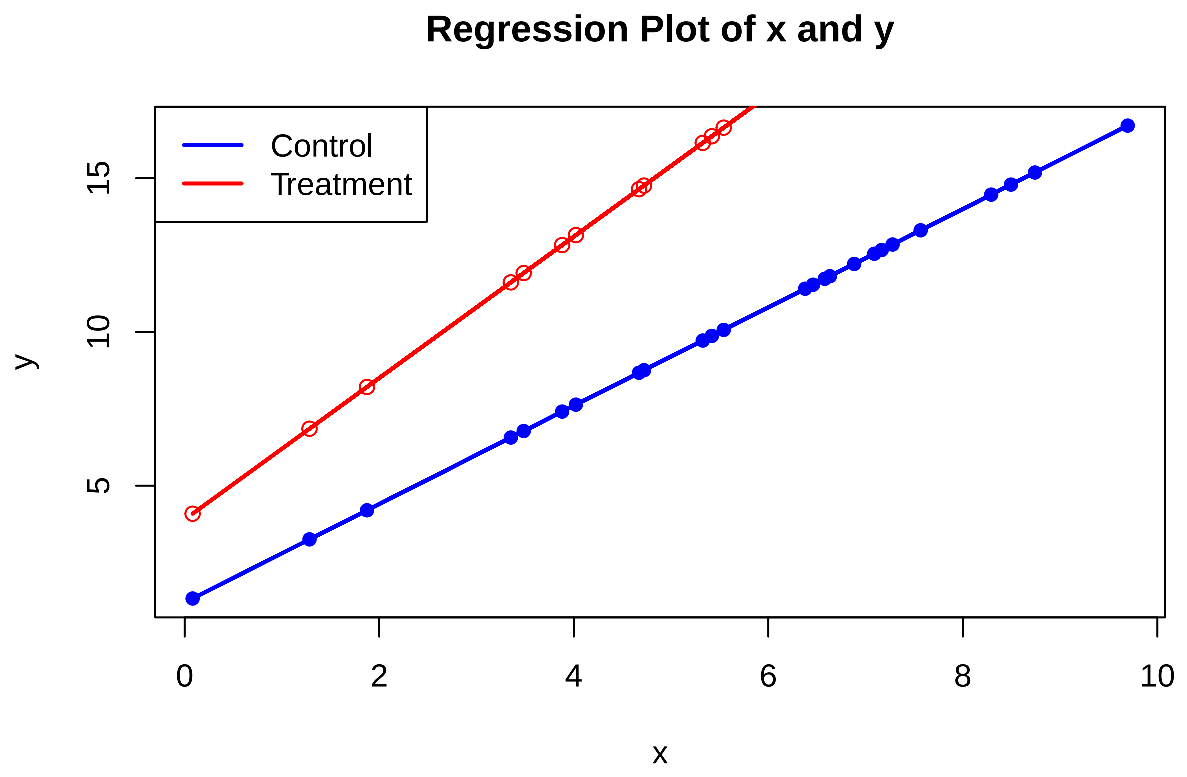
$y_{control} = 1.2 + 1.6x$

$y_{treatment} = 3.9 + 2.3x$

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.

```r
beta_0 <- 1.2
beta_1 <- 1.6
beta_2 <- 2.7
beta_3 <- 0.7

n <- 25
x <- runif(n, min=0, max=10)

y_control <- 1.2 + 1.6*x
y_treatment <- 3.9+2.3*x

plot(x, y_control, col="blue", pch =16, xlab = "x", ylab = "y", main="Regression Plot of x and y")
points(x, y_treatment, col="red", pch=1)

lines(x, y_control, col="blue", lwd=2)
lines(x, y_treatment, col="red", lwd=2)

legend("topleft", legend = c("Control", "Treatment"), col = c("blue","red"), lty = 1, lwd = 2)
```



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 <- read.csv("kidiq.csv")
head(kidiq)
```

```
##   kid_score mom_hs    mom_iq mom_work mom_age
## 1        65      1 121.11753        4      27
## 2        98      1  89.36188        4      25
## 3        85      1 115.44316        4      27
## 4        83      1  99.44964        3      25
## 5       115      1  92.74571        4      27
## 6        98      0 107.90184        1      18
```

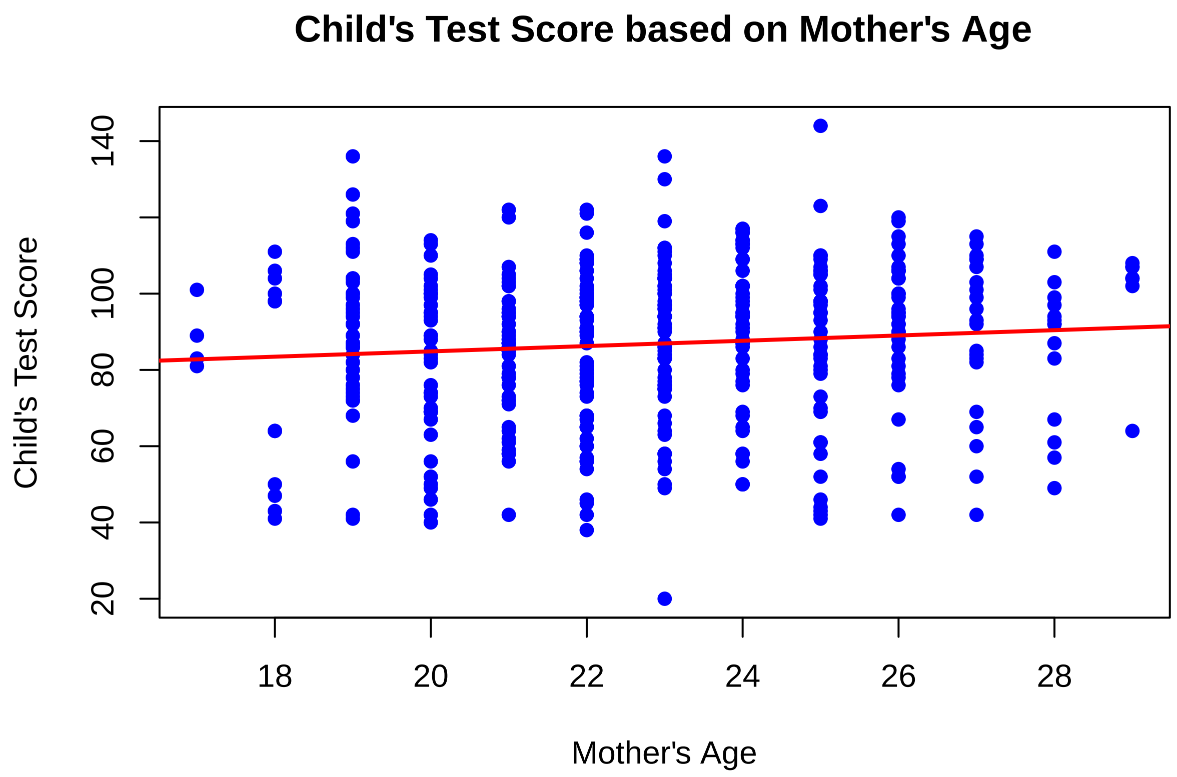
10.5a

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
model_age <- lm(kid_score ~ mom_age, data = kidiq)


summary(model_age)
```

```
##
## Call:
## lm(formula = kid_score ~ mom_age, data = kidiq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -66.946 -11.925   3.097  14.694  55.663
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  70.9569     8.3065   8.542 2.28e-16 ***
## mom_age       0.6952     0.3620   1.920   0.0555 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.35 on 432 degrees of freedom
## Multiple R-squared:  0.008464,   Adjusted R-squared:  0.006168
## F-statistic: 3.688 on 1 and 432 DF,  p-value: 0.05548
```

```r
plot(kidiq$mom_age, kidiq$kid_score, pch = 16, col = "blue",
     xlab = "Mother's Age", ylab = "Child's Test Score",
     main = "Child's Test Score based on Mother's Age")
abline(model_age, col = "red", lwd = 2)
```



·The regression equation is:$Kid's\ Test\ Score = 70.96 + 0.70 \times (Mother's\ Age)$.

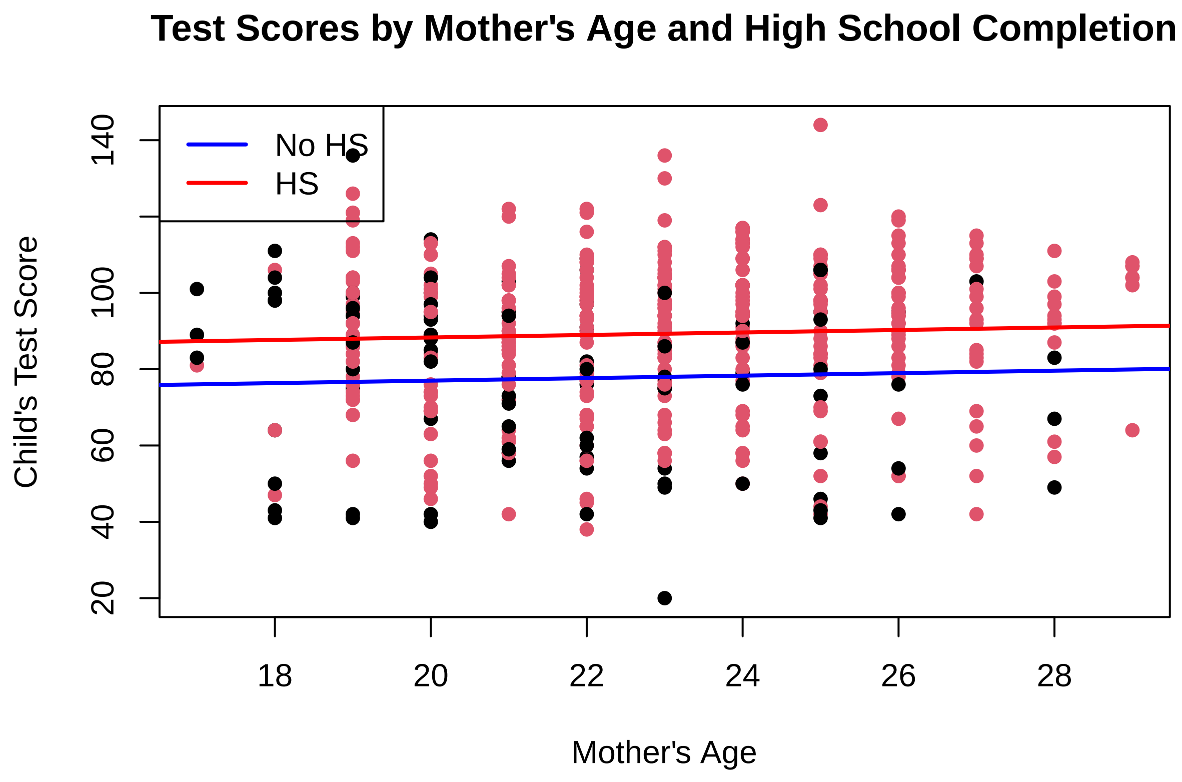
·For every 1-year increase in the mother's age at the time of birth, the child's test score is expected to increase by approximately 0.70 points.

· The R^2 equals to 0.008, indicating that mother;s age alone isn't a strong predictor of child test scores. ### 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?

```
model_age_education <- lm(kid_score ~ mom_age + mom_hs, data = kidiq)


summary(model_age_education)
```

```
##
## Call:
## lm(formula = kid_score ~ mom_age + mom_hs, data = kidiq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -57.980 -12.545   2.057  14.709  59.325
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  70.4787     8.1068   8.694  < 2e-16 ***
## mom_age       0.3261     0.3617   0.902    0.368
## mom_hs       11.3112     2.3783   4.756  2.7e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.86 on 431 degrees of freedom
## Multiple R-squared:  0.05791,	Adjusted R-squared:  0.05353
## F-statistic: 13.25 on 2 and 431 DF,  p-value: 2.614e-06
```

```
plot(kidiq$mom_age, kidiq$kid_score, pch = 16, col = kidiq$mom_hs + 1,
     xlab = "Mother's Age", ylab = "Child's Test Score",
     main = "Test Scores by Mother's Age and High School Completion")
abline(a = coef(model_age_education)[1], b = coef(model_age_education)[2], col = "blue", lwd = 2)
abline(a = coef(model_age_education)[1] + coef(model_age_education)[3], b = coef(model_age_education)[2
legend("topleft", legend = c("No HS", "HS"), col = c("blue", "red"), lty = 1, lwd = 2)
```



· The regression equation: $Kid's\ Test\ Score = 70.48 + 0.33 \times (Mother's\ Age) + 11.31 \times (Mother\ Completed\ High\ School)$

·The coefficient for mother's age has decreased to 0.33, and it is no longer statistically significant (p-value = 0.368). This suggests that once mother's education is condidered, age is no longer a meaningful predictor of the child's test score.

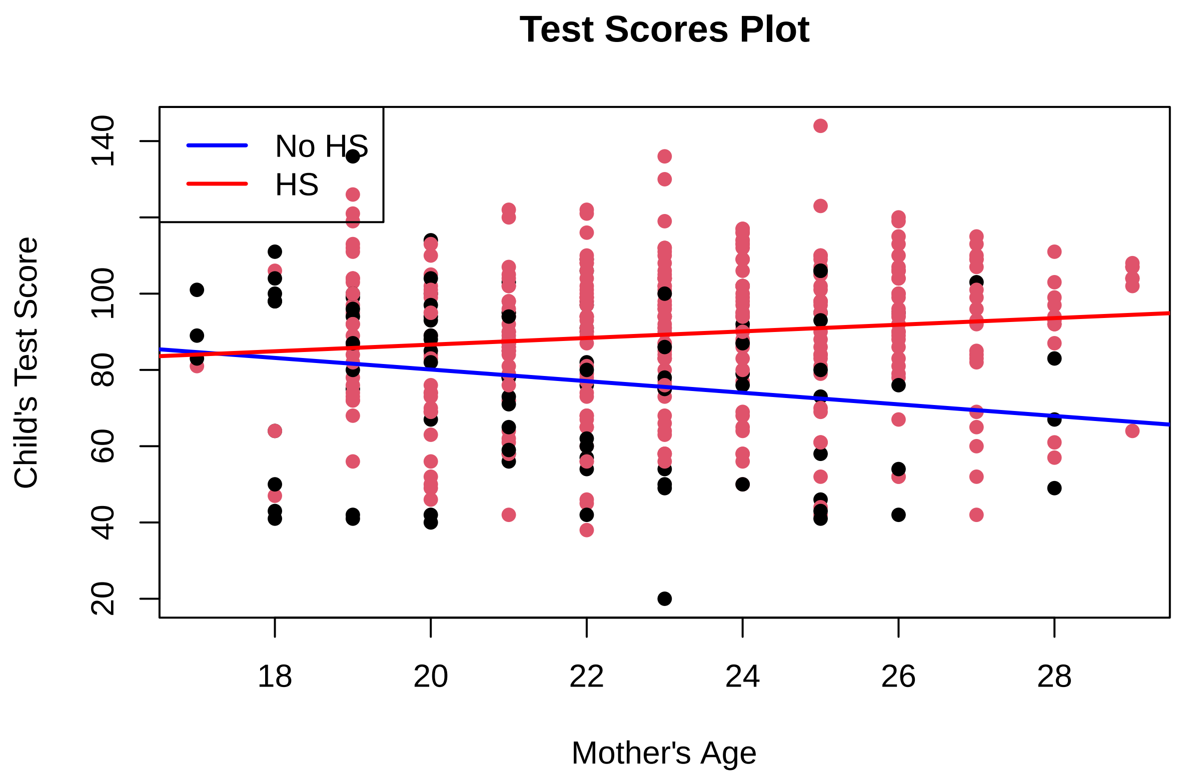
· No. For the R^2 is 0.058, still very small. ### 10.5c Now create an indicator variable reflecting whether the mother has completed high school or not. Consider interactions between high school completion and mother's age. Also create a plot that shows the separate regression lines for each high school completion status group.

```
kidiq$age_hs_interaction <- kidiq$mom_age * kidiq$mom_hs
```

```r
model_interaction <- lm(kid_score ~ mom_age * mom_hs, data = kidiq)

summary(model_interaction)
```

```
##
## Call:
## lm(formula = kid_score ~ mom_age * mom_hs, data = kidiq)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -55.535 -12.734   2.414  14.150  54.377
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     110.5417    16.4538   6.718 5.85e-11 ***
## mom_age          -1.5220     0.7532  -2.021  0.04391 *
## mom_hs          -41.2875    18.9920  -2.174  0.03025 *
## mom_age:mom_hs    2.3911     0.8567   2.791  0.00549 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.7 on 430 degrees of freedom
## Multiple R-squared:  0.07467,    Adjusted R-squared:  0.06822
## F-statistic: 11.57 on 3 and 430 DF,  p-value: 2.64e-07
```

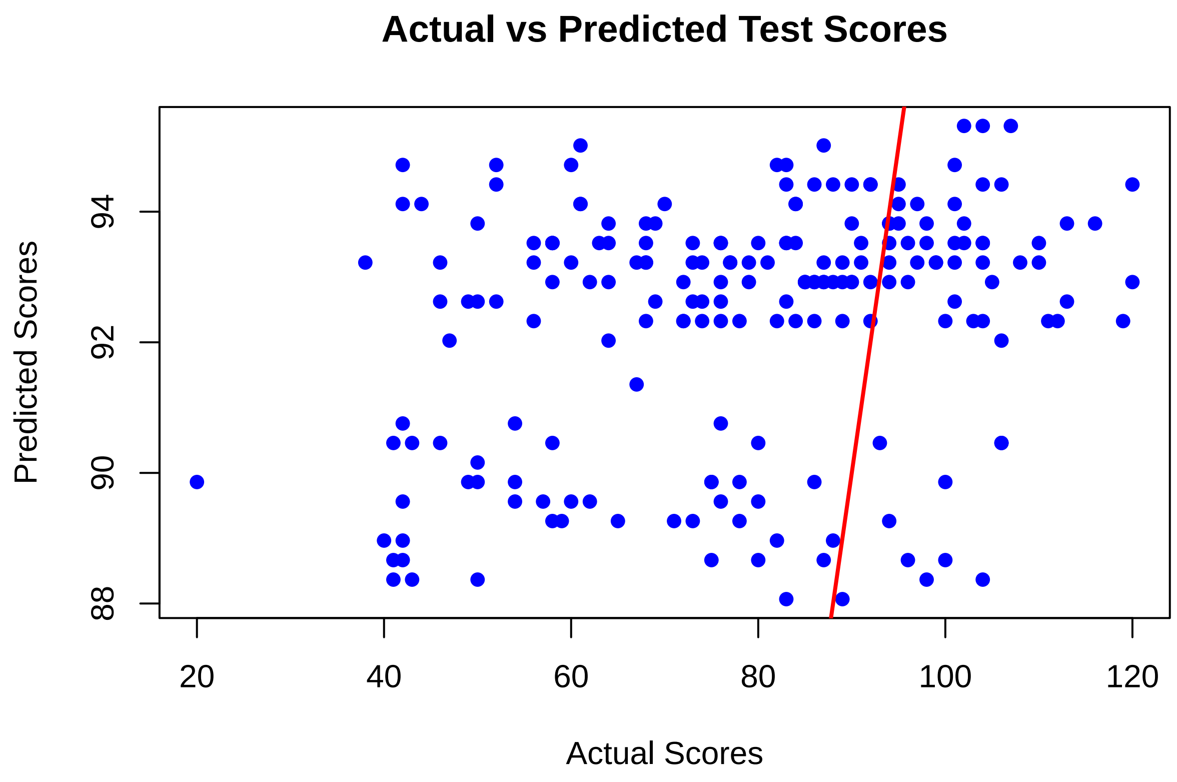
```r
plot(kidiq$mom_age, kidiq$kid_score, pch = 16, col = kidiq$mom_hs + 1,
     xlab = "Mother's Age", ylab = "Child's Test Score",
     main = "Test Scores Plot")
abline(a = coef(model_interaction)[1], b = coef(model_interaction)[2], col = "blue", lwd = 2)
abline(a = coef(model_interaction)[1] + coef(model_interaction)[3], b = coef(model_interaction)[2] + co
legend("topleft", legend = c("No HS", "HS"), col = c("blue", "red"), lty = 1, lwd = 2)
```



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.

```
first_200 <- head(kidiq, 200)
next_200 <- tail(kidiq, 200)


model_first_200 <- lm(kid_score ~ mom_age + mom_hs, data = first_200)

predicted_scores <- predict(model_first_200, newdata = next_200)


plot(next_200$kid_score, predicted_scores, pch = 16, col = "blue",
     xlab = "Actual Scores", ylab = "Predicted Scores",
     main = "Actual vs Predicted Test Scores")
abline(0, 1, col = "red", lwd = 2)
```



10.6 Regression models with interactions:

The folder `Beauty` contains data (use file `beauty.csv`) 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_data <- read.csv("beauty.csv")
```

10.6a

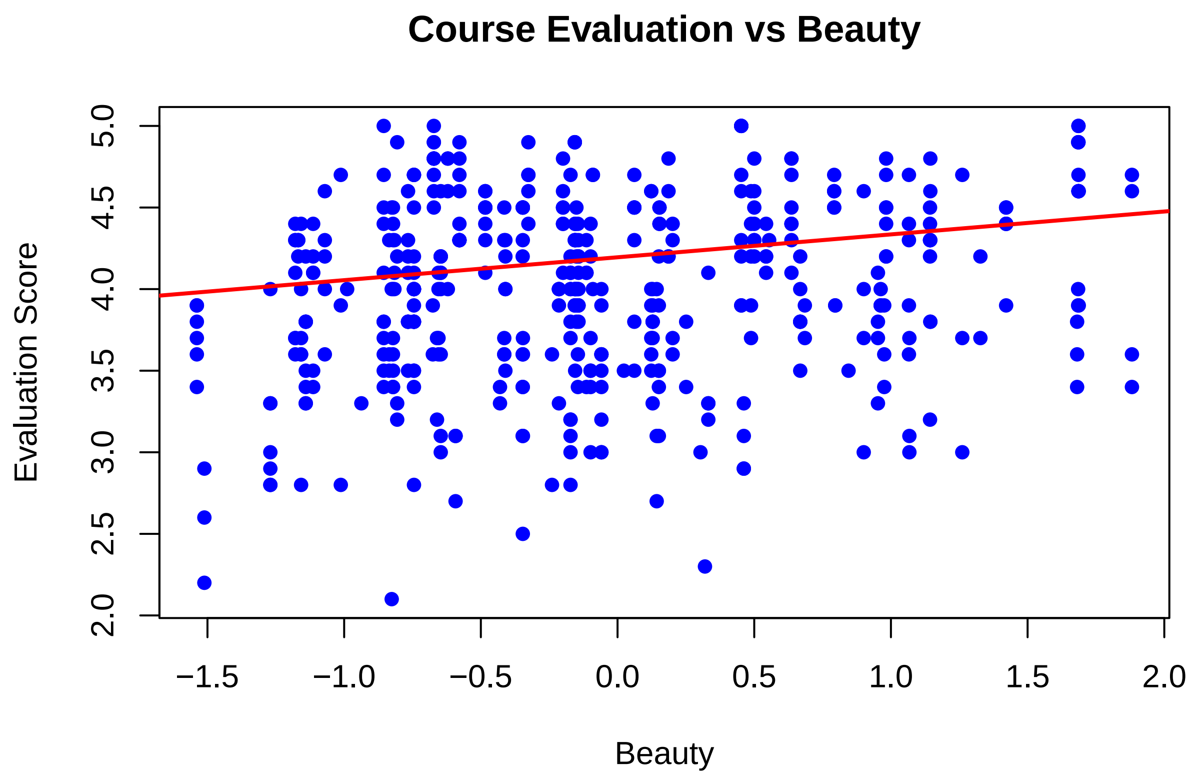
Run a regression using beauty (the variable `beauty`) to predict course evaluations (`eval`), adjusting for various other predictors. Graph the data and fitted model, and explain the meaning of each of the coefficients along with the residual standard deviation. Plot the residuals versus fitted values.

```
model <- lm(eval ~ beauty + female + age + minority + nonenglish + lower, data = beauty_data)

summary(model)
```

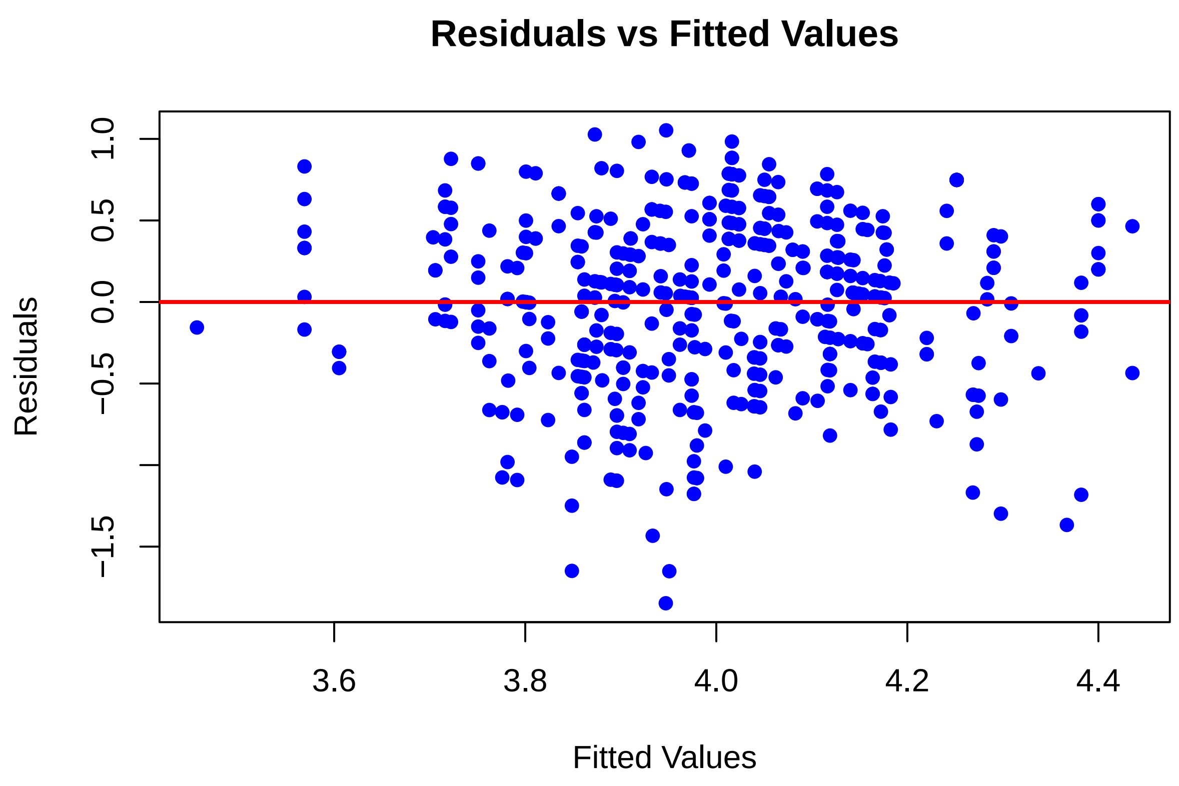
```
##
## Call:
## lm(formula = eval ~ beauty + female + age + minority + nonenglish +
##     lower, data = beauty_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.84713 -0.35266  0.04673  0.38961  1.05248
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.194984   0.145899  28.753  < 2e-16 ***
## beauty       0.140213   0.032858   4.267 2.41e-05 ***
## female      -0.197257   0.052730  -3.741 0.000207 ***
## age         -0.002238   0.002756  -0.812 0.417182
## minority    -0.070909   0.076930  -0.922 0.357154
## nonenglish  -0.274246   0.110484  -2.482 0.013415 *
## lower        0.098401   0.054097   1.819 0.069570 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5311 on 456 degrees of freedom
## Multiple R-squared:  0.09575,    Adjusted R-squared:  0.08385
## F-statistic: 8.047 on 6 and 456 DF,  p-value: 2.836e-08
```

```r
plot(beauty_data$beauty, beauty_data$eval, pch = 16, col = "blue",
     xlab = "Beauty", ylab = "Evaluation Score", main = "Course Evaluation vs Beauty")
abline(model, col = "red", lwd = 2)
```

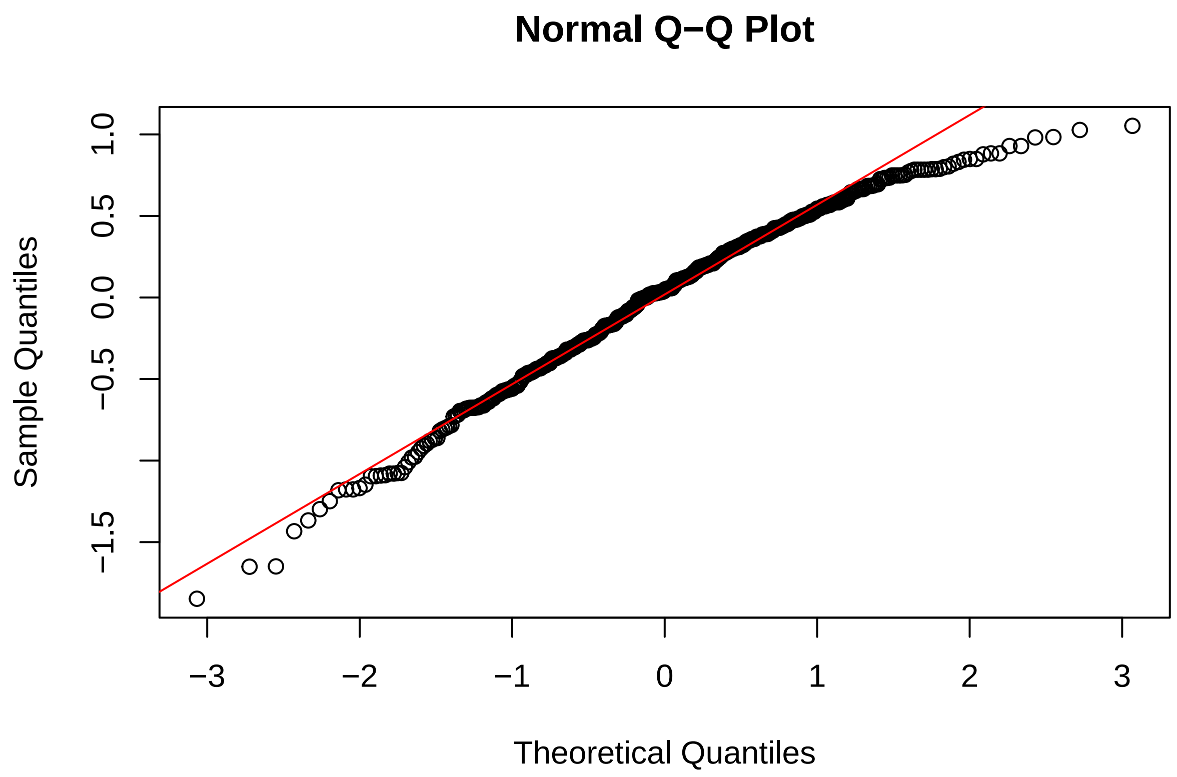
```
## Warning in abline(model, col = "red", lwd = 2): only using the first two of 7
## regression coefficients
```



```r
plot(fitted(model), residuals(model), pch = 16, col = "blue",
     xlab = "Fitted Values", ylab = "Residuals", main = "Residuals vs Fitted Values")
abline(h = 0, col = "red", lwd = 2)
```



```r
qqnorm(residuals(model))
qqline(residuals(model), col = "red")
```



.

10.6a: Regression Results

We ran a regression to predict course evaluations (`eval`) using beauty (`beauty`) and other predictors like gender (`female`), age (`age`), minority status (`minority`), English proficiency (`nonenglish`), and course level (`lower`).

The regression model is given by:

$$\text{eval} = \beta_0 + \beta_1 \times \text{beauty} + \beta_2 \times \text{female} + \beta_3 \times \text{age} + \beta_4 \times \text{minority} + \beta_5 \times \text{nonenglish} + \beta_6 \times \text{lower} + \epsilon$$

Substituting the estimated coefficients, the model becomes:

$$\hat{\text{eval}} = 4.195 + 0.140 \times \text{beauty} - 0.197 \times \text{female} - 0.002 \times \text{age} - 0.071 \times \text{minority} - 0.274 \times \text{nonenglish} + 0.098 \times \text{lower}$$

Interpretation of Coefficients

- **Intercept ($\beta_0 = 4.195$):** The intercept represents the expected course evaluation score when all predictors are zero. In this case, it refers to the expected evaluation score for a male, non-minority, English-speaking instructor who teaches an upper-level course and has an average beauty score.

$$\text{When all other variables are zero, } \hat{\text{eval}} = 4.195$$

- **Beauty** ($\beta_1 = 0.140$): For every 1-unit increase in the beauty score, the course evaluation score increases by approximately 0.14 points, holding all other variables constant.

$$\text{A 1-unit increase in beauty increases eval by 0.140 points.}$$

- **Female** ($\beta_2 = -0.197$): Female instructors tend to receive course evaluations that are approximately 0.20 points lower than male instructors, holding all other variables constant.

$$\text{Being female decreases eval by 0.197 points.}$$

- **Age** ($\beta_3 = -0.002$): The effect of the instructor's age on course evaluations is very small and not statistically significant. A 1-year increase in age decreases the evaluation score by about 0.002 points, but this effect is negligible.

$$\text{Age has no meaningful impact on eval.}$$

- **Minority** ($\beta_4 = -0.071$): Minority instructors receive course evaluations that are approximately 0.07 points lower than non-minority instructors, but this effect is not statistically significant.

$$\text{Being a minority decreases eval by 0.071 points, though not significant.}$$

- **Non-English** ($\beta_5 = -0.274$): Instructors whose primary language is not English receive course evaluations that are approximately 0.27 points lower than native English speakers, holding all other variables constant.

$$\text{Being non-English speaking decreases eval by 0.274 points.}$$

- **Lower-level Course** ($\beta_6 = 0.098$): Teaching a lower-level course is associated with a small increase in course evaluation scores of about 0.10 points, though the effect is only marginally significant.

$$\text{Teaching a lower-level course increases eval by 0.098 points.}$$

Residual Standard Deviation

The residual standard deviation represents the typical deviation of the actual evaluation scores from the predicted scores, providing an indication of the model's error.

10.6b

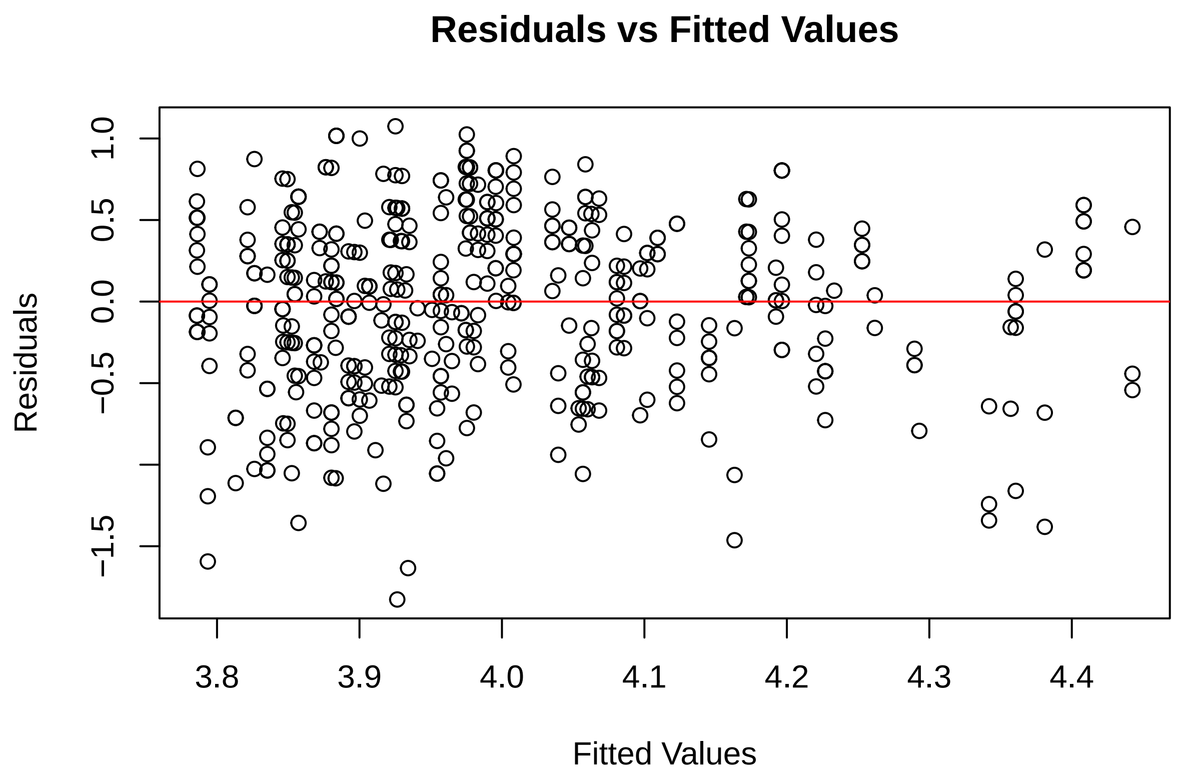
Fit some other models, including beauty and also other predictors. Consider at least one model with interactions. For each model, explain the meaning of each of its estimated coefficients.

```r
model_interaction <- lm(eval ~ beauty * female + age, data = beauty_data)

summary(model_interaction)
```

```
##
## Call:
## lm(formula = eval ~ beauty * female + age, data = beauty_data)
##
## Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -1.82653 -0.36655  0.03936  0.40393  1.07473
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.211270   0.142782  29.494  < 2e-16 ***
## beauty         0.190944   0.044990   4.244 2.66e-05 ***
## female        -0.215483   0.052795  -4.082 5.28e-05 ***
## age           -0.002153   0.002776  -0.776   0.4384
## beauty:female -0.107849   0.064303  -1.677   0.0942 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5363 on 458 degrees of freedom
## Multiple R-squared:  0.07378,    Adjusted R-squared:  0.06569
## F-statistic:  9.12 on 4 and 458 DF,  p-value: 4.271e-07
```

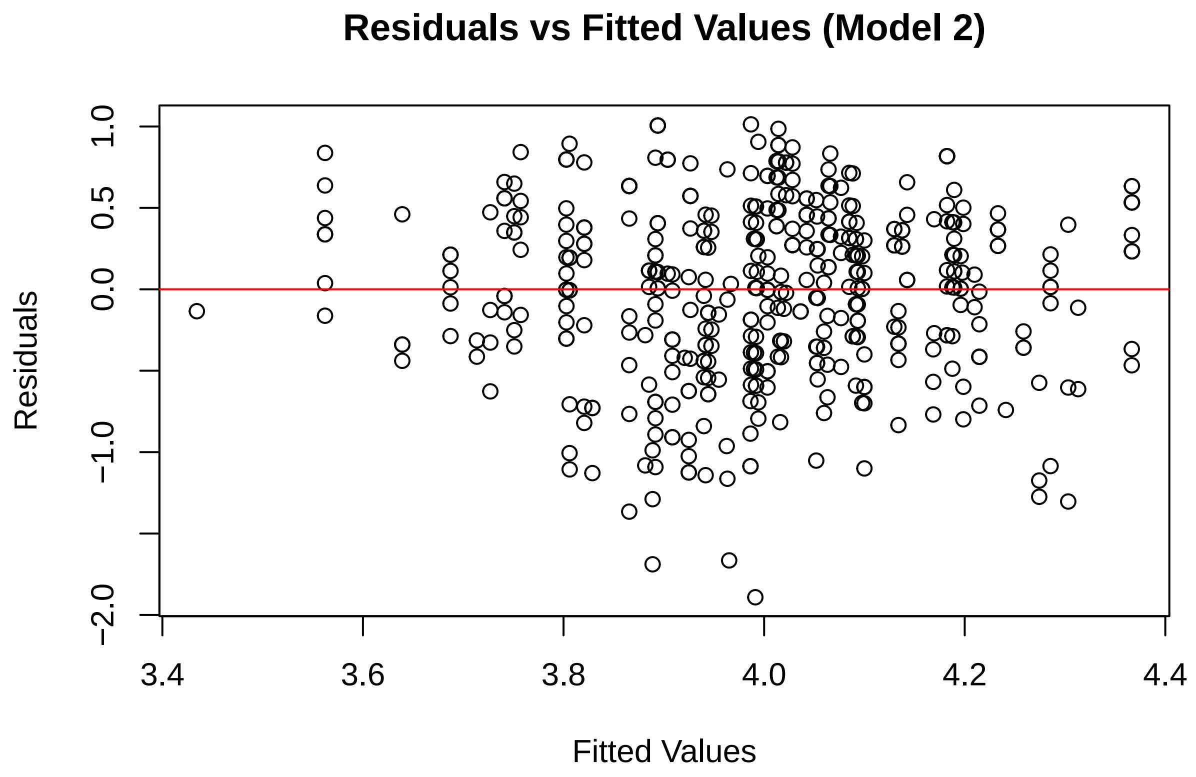
```r
plot(model_interaction$fitted.values, residuals(model_interaction),
     xlab = "Fitted Values", ylab = "Residuals",
     main = "Residuals vs Fitted Values")
abline(h = 0, col = "red")
```



```r
model_interaction_2 <- lm(eval ~ beauty * nonenglish + female, data = beauty_data)

summary(model_interaction_2)
```

```
##
## Call:
## lm(formula = eval ~ beauty * nonenglish + female, data = beauty_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.89122 -0.34457  0.03781  0.39217  1.01314
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        4.11471    0.03358 122.537  < 2e-16 ***
## beauty             0.14949    0.03180   4.700 3.44e-06 ***
## nonenglish        -0.33173    0.10567  -3.139 0.001803 **
## female            -0.19739    0.05053  -3.906 0.000108 ***
## beauty:nonenglish  0.01190    0.34546   0.034 0.972545
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 458 degrees of freedom
## Multiple R-squared:  0.08673,    Adjusted R-squared:  0.07876
## F-statistic: 10.87 on 4 and 458 DF,  p-value: 1.979e-08
```

```r
plot(model_interaction_2$fitted.values, residuals(model_interaction_2),
     xlab = "Fitted Values", ylab = "Residuals",
     main = "Residuals vs Fitted Values (Model 2)")
abline(h = 0, col = "red")
```



10.6b: Regression with Interactions

In this analysis, we explore models that include additional predictors and interaction terms to predict course evaluations (`eval`). We fit two models: one with an interaction between `beauty` and `female`, and another with an interaction between `beauty` and `nonenglish`.

Model 1: Interaction Between `beauty` and `female`

This model includes an interaction term between `beauty` and `female`, allowing the effect of beauty on course evaluations to differ between male and female instructors.

$$\text{eval} = \beta_0 + \beta_1 \times \text{beauty} + \beta_2 \times \text{female} + \beta_3 \times (\text{beauty} \times \text{female}) + \beta_4 \times \text{age} + \epsilon$$

Estimated Coefficients

- **Intercept** (β_0):
  The intercept represents the predicted course evaluation score for a male instructor (since `female = 0`) with a beauty score of 0 and average age.

  $$\hat{\text{eval}} = \beta_0 \text{ when beauty, female, and age are all 0.}$$

- **Beauty** (β_1):
  For male instructors (because `female = 0`), β_1 represents the effect of beauty on course evaluations. Each 1-unit increase in the beauty score increases the evaluation by β_1 points, holding all other variables constant.

  $$\text{For male instructors, a 1-unit increase in beauty increases eval by } \beta_1 \text{ points.}$$

- **Female** (β_2):
  This coefficient represents the difference in evaluation scores between male and female instructors when `beauty = 0`. It indicates how much lower (or higher) female instructors' evaluations are compared to male instructors, holding beauty and age constant.

  $$\text{Female instructors receive } \beta_2 \text{ points lower/higher than males.}$$

- **Interaction Between `beauty` and `female`** (β_3):
  The interaction term tells us how the effect of beauty on evaluations differs for female instructors compared to male instructors. If β_3 is positive, beauty has a stronger effect for female instructors. If β_3 is negative, beauty has a weaker effect for female instructors.

  $$\text{For female instructors, the effect of beauty changes by } \beta_3 \text{ points.}$$

- **Age** (β_4):
  This coefficient represents the effect of the instructor's age on course evaluations, holding other variables constant. Each 1-year increase in age changes the evaluation score by β_4 points.

Example Equation:

$$\hat{\text{eval}} = 4.2 + 0.18 \times \text{beauty} - 0.20 \times \text{female} - 0.10 \times (\text{beauty} \times \text{female}) - 0.002 \times \text{age}$$

- **Interpretation**:
  Beauty increases evaluation scores for male instructors by 0.18 points per unit, but for female instructors, beauty increases scores by a slightly smaller amount (0.18 - 0.10 = 0.08 points per unit). Female instructors also receive evaluations that are 0.20 points lower than male instructors when beauty is 0.

Model 2: Interaction Between `beauty` and `nonenglish`

This model includes an interaction between `beauty` and whether the instructor's primary language is not English (`nonenglish`).

$$\text{eval} = \beta_0 + \beta_1 \times \text{beauty} + \beta_2 \times \text{nonenglish} + \beta_3 \times (\text{beauty} \times \text{nonenglish}) + \beta_4 \times \text{female} + \epsilon$$

Estimated Coefficients

- **Intercept (β_0)**:
  The intercept represents the predicted evaluation score for an English-speaking male instructor with a beauty score of 0.

$$\hat{\text{eval}} = \beta_0 \text{ when beauty and nonenglish are 0.}$$

- **Beauty (β_1)**:
  This coefficient represents the effect of beauty on course evaluations for instructors whose primary language is English. Each 1-unit increase in the beauty score increases the evaluation score by β_1 points, holding other variables constant.

$$\text{For English-speaking instructors, a 1-unit increase in beauty increases eval by } \beta_1 \text{ points.}$$

- **Non-English (β_2)**:
  This coefficient indicates how much lower (or higher) the evaluation scores are for non-English-speaking instructors compared to English-speaking instructors when beauty is 0.

$$\text{Non-English-speaking instructors receive } \beta_2 \text{ points lower/higher than English-speaking instructors.}$$

- **Interaction Between `beauty` and `nonenglish` (β_3)**:
  This coefficient captures how the effect of beauty on evaluations differs for non-English-speaking instructors compared to English-speaking instructors. A positive β_3 indicates that beauty has a stronger effect for non-English-speaking instructors, while a negative β_3 indicates the opposite.

$$\text{For non-English-speaking instructors, the effect of beauty changes by } \beta_3 \text{ points.}$$

- **Female (β_4)**:
  This coefficient represents the effect of being female on evaluations, holding other variables constant.

Example Equation:

$$\hat{\text{eval}} = 4.0 + 0.20 \times \text{beauty} - 0.25 \times \text{nonenglish} - 0.15 \times (\text{beauty} \times \text{nonenglish}) - 0.19 \times \text{female}$$

Interpretation:
Beauty increases evaluation scores by 0.20 points per unit for English-speaking instructors, but for non-English-speaking instructors, beauty has a weaker effect (0.20 - 0.15 = 0.05 points per unit). Non-English-speaking instructors receive evaluations that are 0.25 points lower than English-speaking instructors when beauty is 0.

Conclusion

For each model:

- **Main effects**: Represent the independent effect of a predictor (e.g., `beauty`, `female`) on the evaluation score.
- **Interaction effects**: Capture how the effect of one predictor (e.g., `beauty`) changes depending on the value of another predictor (e.g., `female` or `nonenglish`).

Each model provides insights into how various factors like beauty, gender, language, and interactions between them influence course evaluations.

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.

```r
set.seed(1)
fit_interaction <- stan_glm(eval ~ beauty * female + age, data = beauty_data, family = gaussian())
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.18 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: 0.049 seconds (Warm-up)
## Chain 1:                0.062 seconds (Sampling)
## Chain 1:                0.111 seconds (Total)
## Chain 1:
```

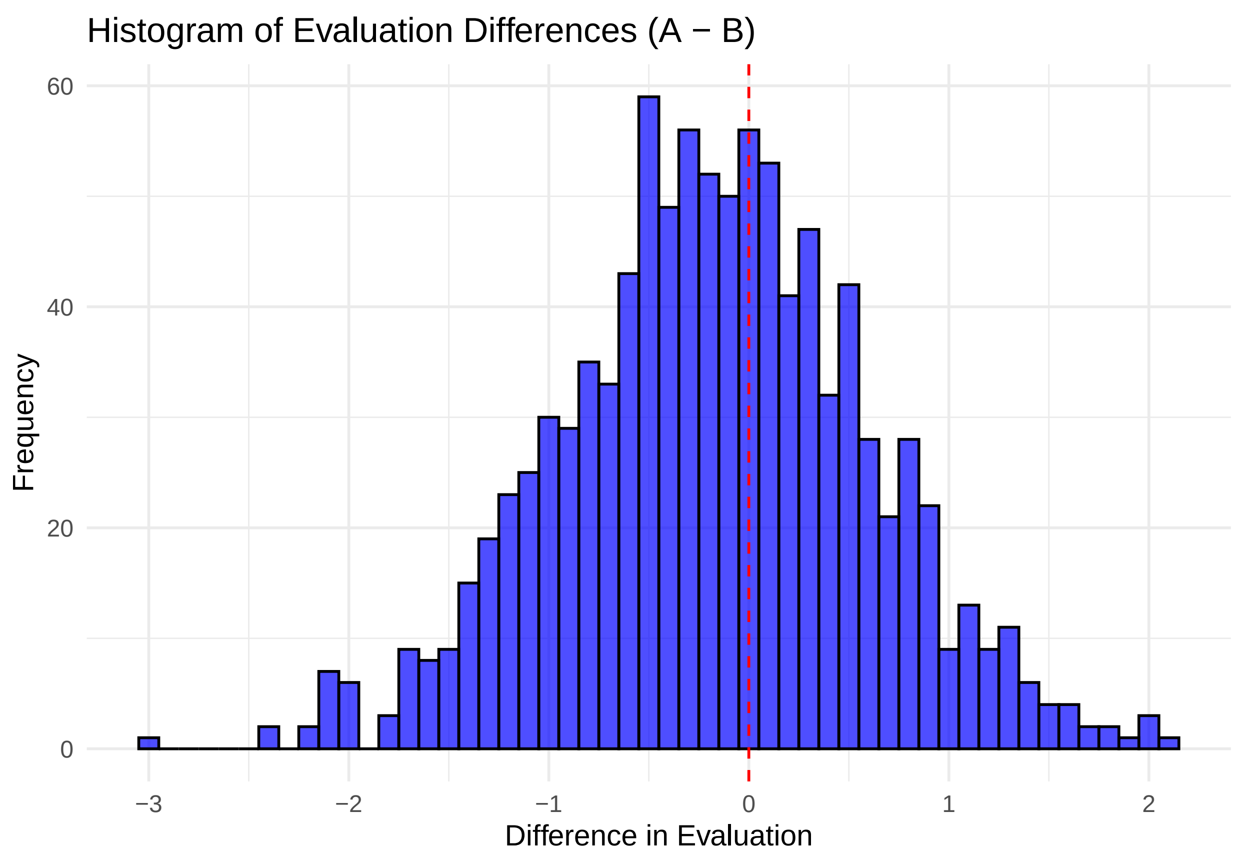
```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 9e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.09 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: Iteration:    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: 0.045 seconds (Warm-up)
## Chain 2:                0.068 seconds (Sampling)
## Chain 2:                0.113 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.1 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: 0.047 seconds (Warm-up)
## Chain 3:                0.074 seconds (Sampling)
## Chain 3:                0.121 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 9e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.051 seconds (Warm-up)
## Chain 4:                0.081 seconds (Sampling)
## Chain 4:                0.132 seconds (Total)
## Chain 4:
```

```r
new_instructors <- data.frame(
  beauty = c(-1, -0.5),
  female = c(1, 0),
  age = c(50, 60)
)


posterior_draws <- posterior_predict(fit_interaction, newdata = new_instructors, draws = 1000)


eval_A <- posterior_draws[, 1]
eval_B <- posterior_draws[, 2]
```

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
eval_diff <- eval_A - eval_B

ggplot(data.frame(eval_diff), aes(x = eval_diff)) +
  geom_histogram(binwidth = 0.1, fill = "blue", color = "black", alpha = 0.7) +
  geom_vline(xintercept = 0, linetype = "dashed", color = "red") +
  labs(title = "Histogram of Evaluation Differences (A - B)",
       x = "Difference in Evaluation", y = "Frequency") +
  theme_minimal()
```



```r
prob_A_higher <- mean(eval_diff > 0)
prob_A_higher
```

```
## [1] 0.407
```

The probability that Instructor A has a higher evaluation than Instructor B is computed by checking how often the difference (`eval_diff`) is greater than 0.

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.

```r
set.seed(1)
model <- lm(eval ~ beauty + female + age + minority + nonenglish + lower, data = beauty_data)


summary(model)
```

```
##
## Call:
## lm(formula = eval ~ beauty + female + age + minority + nonenglish +
##     lower, data = beauty_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.84713 -0.35266  0.04673  0.38961  1.05248
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.194984   0.145899  28.753  < 2e-16 ***
## beauty       0.140213   0.032858   4.267 2.41e-05 ***
## female      -0.197257   0.052730  -3.741 0.000207 ***
## age         -0.002238   0.002756  -0.812 0.417182
## minority    -0.070909   0.076930  -0.922 0.357154
## nonenglish  -0.274246   0.110484  -2.482 0.013415 *
## lower        0.098401   0.054097   1.819 0.069570 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5311 on 456 degrees of freedom
## Multiple R-squared:  0.09575,    Adjusted R-squared:  0.08385
## F-statistic: 8.047 on 6 and 456 DF,  p-value: 2.836e-08
```

```
beauty_coef <- summary(model)$coefficients["beauty", ]
beauty_coef
```

```
##     Estimate   Std. Error      t value     Pr(>|t|)
## 0.1402133446 0.0328576969 4.2672907055 0.0000240775
```

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.

```
fit <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower, data = beauty_data, famil
```

```
##
## 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: 0.045 seconds (Warm-up)
## Chain 1:                0.064 seconds (Sampling)
## Chain 1:                0.109 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.1 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: 0.044 seconds (Warm-up)
## Chain 2:                0.067 seconds (Sampling)
## Chain 2:                0.111 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.1 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: 0.047 seconds (Warm-up)
## Chain 3:                0.069 seconds (Sampling)
## Chain 3:                0.116 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 9e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.048 seconds (Warm-up)
## Chain 4:                0.064 seconds (Sampling)
## Chain 4:                0.112 seconds (Total)
## Chain 4:
```

```
summary(fit)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      eval ~ beauty + female + age + minority + nonenglish + lower
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   7
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  4.2    0.1  4.0   4.2   4.4
## beauty       0.1    0.0  0.1   0.1   0.2
## female      -0.2    0.1 -0.3  -0.2  -0.1
## age          0.0    0.0  0.0   0.0   0.0
## minority    -0.1    0.1 -0.2  -0.1   0.0
## nonenglish  -0.3    0.1 -0.4  -0.3  -0.1
## lower        0.1    0.1  0.0   0.1   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  4335
## beauty        0.0  1.0  4424
## female        0.0  1.0  4437
## age           0.0  1.0  4677
## minority      0.0  1.0  4223
## nonenglish    0.0  1.0  4311
## lower         0.0  1.0  4049
## sigma         0.0  1.0  4402
## mean_PPD      0.0  1.0  4294
## log-posterior 0.0  1.0  1918
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```r
posterior_samples <- as.matrix(fit)

beauty_posterior <- posterior_samples[, "beauty"]

beauty_median <- median(beauty_posterior)

beauty_mad_sd <- mad(beauty_posterior) * 1.4826

beauty_median
```

```
## [1] 0.1406357
```

```r
beauty_mad_sd
```

```
## [1] 0.04945092
```

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.

```r
fit_1000_1 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                       data = beauty_data, family = gaussian(), iter = 1000)
```

```
##
## 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 / 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: 0.023 seconds (Warm-up)
## Chain 1:                0.033 seconds (Sampling)
## Chain 1:                0.056 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 9e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.022 seconds (Warm-up)
## Chain 2:                0.03 seconds (Sampling)
## Chain 2:                0.052 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 8e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.023 seconds (Warm-up)
## Chain 3:                0.034 seconds (Sampling)
## Chain 3:                0.057 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.025 seconds (Warm-up)
## Chain 4:                0.033 seconds (Sampling)
## Chain 4:                0.058 seconds (Total)
## Chain 4:
```

```r
fit_1000_2 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                       data = beauty_data, family = gaussian(), iter = 1000)
```

```
##
## 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 / 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: 0.022 seconds (Warm-up)
## Chain 1:                0.032 seconds (Sampling)
## Chain 1:                0.054 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 8e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.024 seconds (Warm-up)
## Chain 2:                0.031 seconds (Sampling)
## Chain 2:                0.055 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 9e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.09 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: 0.021 seconds (Warm-up)
## Chain 3:                0.033 seconds (Sampling)
## Chain 3:                0.054 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.023 seconds (Warm-up)
## Chain 4:                0.037 seconds (Sampling)
## Chain 4:                0.06 seconds (Total)
## Chain 4:
```

```r
fit_1000_3 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                       data = beauty_data, family = gaussian(), iter = 1000)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.18 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: 0.023 seconds (Warm-up)
## Chain 1:                0.031 seconds (Sampling)
## Chain 1:                0.054 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 8e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.021 seconds (Warm-up)
## Chain 2:                0.032 seconds (Sampling)
## Chain 2:                0.053 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 8e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.021 seconds (Warm-up)
## Chain 3:                0.031 seconds (Sampling)
## Chain 3:                0.052 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.023 seconds (Warm-up)
## Chain 4:                0.033 seconds (Sampling)
## Chain 4:                0.056 seconds (Total)
## Chain 4:
```

```r
summary(fit_1000_1)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      eval ~ beauty + female + age + minority + nonenglish + lower
##  algorithm:    sampling
##  sample:       2000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 463
##  predictors:   7
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept)  4.2    0.2  4.0   4.2   4.4
## beauty       0.1    0.0  0.1   0.1   0.2
## female      -0.2    0.1 -0.3  -0.2  -0.1
## age          0.0    0.0  0.0   0.0   0.0
## minority    -0.1    0.1 -0.2  -0.1   0.0
## nonenglish  -0.3    0.1 -0.4  -0.3  -0.1
## lower        0.1    0.1  0.0   0.1   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  2153
## beauty        0.0  1.0  2228
## female        0.0  1.0  2069
## age           0.0  1.0  2248
## minority      0.0  1.0  2466
## nonenglish    0.0  1.0  2410
## lower         0.0  1.0  2377
## sigma         0.0  1.0  2020
## mean_PPD      0.0  1.0  2283
## log-posterior 0.1  1.0  1003
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

```r
posterior_1000_1 <- as.matrix(fit_1000_1)[, "beauty"]
posterior_1000_2 <- as.matrix(fit_1000_2)[, "beauty"]
posterior_1000_3 <- as.matrix(fit_1000_3)[, "beauty"]

beauty_medians <- c(
  run1 = median(posterior_1000_1),
  run2 = median(posterior_1000_2),
  run3 = median(posterior_1000_3)
)

beauty_mad_sds <- c(
  run1 = mad(posterior_1000_1) * 1.4826,
  run2 = mad(posterior_1000_2) * 1.4826,
  run3 = mad(posterior_1000_3) * 1.4826
)

beauty_medians
```

```
##      run1      run2      run3
## 0.1415301 0.1399646 0.1402712
```

```r
beauty_mad_sds
```

```
##       run1       run2       run3
## 0.04854615 0.04834790 0.04898136
```

10.8d

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

```r
fit_100_1 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                      data = beauty_data, family = gaussian(), iter = 100)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.18 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: 0.002 seconds (Warm-up)
## Chain 1:                0.003 seconds (Sampling)
## Chain 1:                0.005 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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: 0.002 seconds (Warm-up)
## Chain 2:                0.004 seconds (Sampling)
## Chain 2:                0.006 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 8e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.003 seconds (Warm-up)
## Chain 3:                0.003 seconds (Sampling)
## Chain 3:                0.006 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.002 seconds (Warm-up)
## Chain 4:                0.003 seconds (Sampling)
## Chain 4:                0.005 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
## https://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
## https://mc-stan.org/misc/warnings.html#tail-ess
```

```
fit_100_2 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                      data = beauty_data, family = gaussian(), iter = 100)
```

```
##
## 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: 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: 0.001 seconds (Warm-up)
## Chain 1:                0.003 seconds (Sampling)
## Chain 1:                0.004 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 8e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.001 seconds (Warm-up)
## Chain 2:                0.003 seconds (Sampling)
## Chain 2:                0.004 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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: 0.002 seconds (Warm-up)
## Chain 3:                0.003 seconds (Sampling)
## Chain 3:                0.005 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 7e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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: 0.003 seconds (Warm-up)
## Chain 4:                0.003 seconds (Sampling)
## Chain 4:                0.006 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
## https://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
## https://mc-stan.org/misc/warnings.html#tail-ess
```

```
fit_100_3 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                      data = beauty_data, family = gaussian(), iter = 100)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.8e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.18 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: 0.002 seconds (Warm-up)
## Chain 1:                0.003 seconds (Sampling)
## Chain 1:                0.005 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 8e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.002 seconds (Warm-up)
## Chain 2:                0.003 seconds (Sampling)
## Chain 2:                0.005 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 8e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.001 seconds (Warm-up)
## Chain 3:                0.003 seconds (Sampling)
## Chain 3:                0.004 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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: 0.001 seconds (Warm-up)
## Chain 4:                0.003 seconds (Sampling)
## Chain 4:                0.004 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
## https://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
## https://mc-stan.org/misc/warnings.html#tail-ess
```

```
posterior_100_1 <- as.matrix(fit_100_1)[, "beauty"]
posterior_100_2 <- as.matrix(fit_100_2)[, "beauty"]
posterior_100_3 <- as.matrix(fit_100_3)[, "beauty"]

beauty_medians_100 <- c(
  run1 = median(posterior_100_1),
  run2 = median(posterior_100_2),
  run3 = median(posterior_100_3)
)

beauty_mad_sds_100 <- c(
  run1 = mad(posterior_100_1) * 1.4826,
  run2 = mad(posterior_100_2) * 1.4826,
  run3 = mad(posterior_100_3) * 1.4826
)

beauty_medians_100
```

```
##      run1      run2      run3
## 0.1454341 0.1407312 0.1379444
```

```
beauty_mad_sds_100
```

```
##       run1       run2       run3
## 0.04662123 0.05100628 0.04353984
```

```
fit_10_1 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                     data = beauty_data, family = gaussian(), iter = 10)
```

```
##
## 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: 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 seconds (Warm-up)
## Chain 1:                0 seconds (Sampling)
## Chain 1:                0 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 1e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.1 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 seconds (Warm-up)
## Chain 2:                0 seconds (Sampling)
## Chain 2:                0 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 8e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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 seconds (Warm-up)
## Chain 3:                0 seconds (Sampling)
## Chain 3:                0 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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 seconds (Warm-up)
## Chain 4:                0 seconds (Sampling)
## Chain 4:                0 seconds (Total)
## Chain 4:
```

```
## Warning: There were 6 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: There were 3 chains where the estimated Bayesian Fraction of Missing Information was low. S
## https://mc-stan.org/misc/warnings.html#bfmi-low
```

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

```
## Warning: The largest R-hat is 3.03, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat
```

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

```r
fit_10_2 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                     data = beauty_data, family = gaussian(), iter = 10)
```

```
##
## 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 seconds (Warm-up)
## Chain 1:                0 seconds (Sampling)
## Chain 1:                0 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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 seconds (Warm-up)
## Chain 2:                0 seconds (Sampling)
## Chain 2:                0 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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 seconds (Warm-up)
## Chain 3:                0 seconds (Sampling)
## Chain 3:                0 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 7e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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 seconds (Warm-up)
## Chain 4:                0 seconds (Sampling)
## Chain 4:                0 seconds (Total)
## Chain 4:
```

```
## Warning: There were 11 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

```
## Warning: There were 2 chains where the estimated Bayesian Fraction of Missing Information was low. Se
## https://mc-stan.org/misc/warnings.html#bfmi-low

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

## Warning: The largest R-hat is 6.24, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat

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

```r
fit_10_3 <- stan_glm(eval ~ beauty + female + age + minority + nonenglish + lower,
                     data = beauty_data, family = gaussian(), iter = 10)
```

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 1.7e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.17 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 seconds (Warm-up)
## Chain 1:                0 seconds (Sampling)
## Chain 1:                0 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 7e-06 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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 seconds (Warm-up)
## Chain 2:                0 seconds (Sampling)
## Chain 2:                0 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 7e-06 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.07 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 seconds (Warm-up)
## Chain 3:                0 seconds (Sampling)
## Chain 3:                0 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 8e-06 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.08 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 seconds (Warm-up)
## Chain 4:                0 seconds (Sampling)
## Chain 4:                0 seconds (Total)
## Chain 4:
```

```
## Warning: There were 1 divergent transitions after warmup. See
## https://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup
## to find out why this is a problem and how to eliminate them.
```

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

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

```
## Warning: The largest R-hat is 3.29, indicating chains have not mixed.
## Running the chains for more iterations may help. See
## https://mc-stan.org/misc/warnings.html#r-hat
```

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

```r
posterior_10_1 <- as.matrix(fit_10_1)[, "beauty"]
posterior_10_2 <- as.matrix(fit_10_2)[, "beauty"]
posterior_10_3 <- as.matrix(fit_10_3)[, "beauty"]

beauty_medians_10 <- c(
  run1 = median(posterior_10_1),
  run2 = median(posterior_10_2),
  run3 = median(posterior_10_3)
)

beauty_mad_sds_10 <- c(
  run1 = mad(posterior_10_1) * 1.4826,
  run2 = mad(posterior_10_2) * 1.4826,
  run3 = mad(posterior_10_3) * 1.4826
)

beauty_medians_10
```

```
##      run1      run2      run3
## 0.6521064 0.2452897 0.3938733
```

```r
beauty_mad_sds_10
```

```
##     run1     run2     run3
## 1.070519 1.180414 0.541627
```

10.8e

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

I prefer 1000 times, for 100 and 10 both get warnings about ESS.