HW 1

Reese Mullen

9-12-24

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.

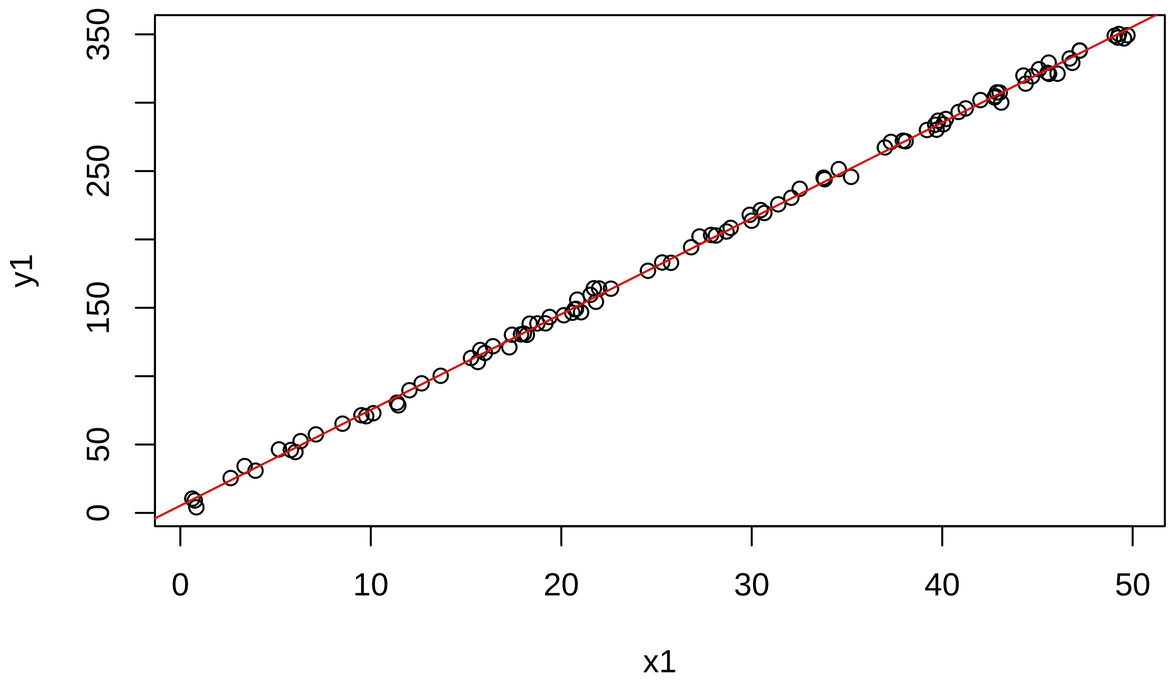
```
x1<-runif(100,0,50)
e1<-rnorm(100,0,3)
y1<-5+7*x1+e1
lm_1<-lm(y1~x1)
summary(lm_1)
```

```
##
## Call:
## lm(formula = y1 ~ x1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.1774 -1.9100  0.2319  2.1306  6.8155
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.36334    0.68745   7.802  6.7e-12 ***
## x1           7.00533    0.02225 314.783  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.174 on 98 degrees of freedom
## Multiple R-squared:  0.999,  Adjusted R-squared:  0.999
## F-statistic: 9.909e+04 on 1 and 98 DF,  p-value: < 2.2e-16
```

7.2b

Graph a scatterplot of the data and the regression line.

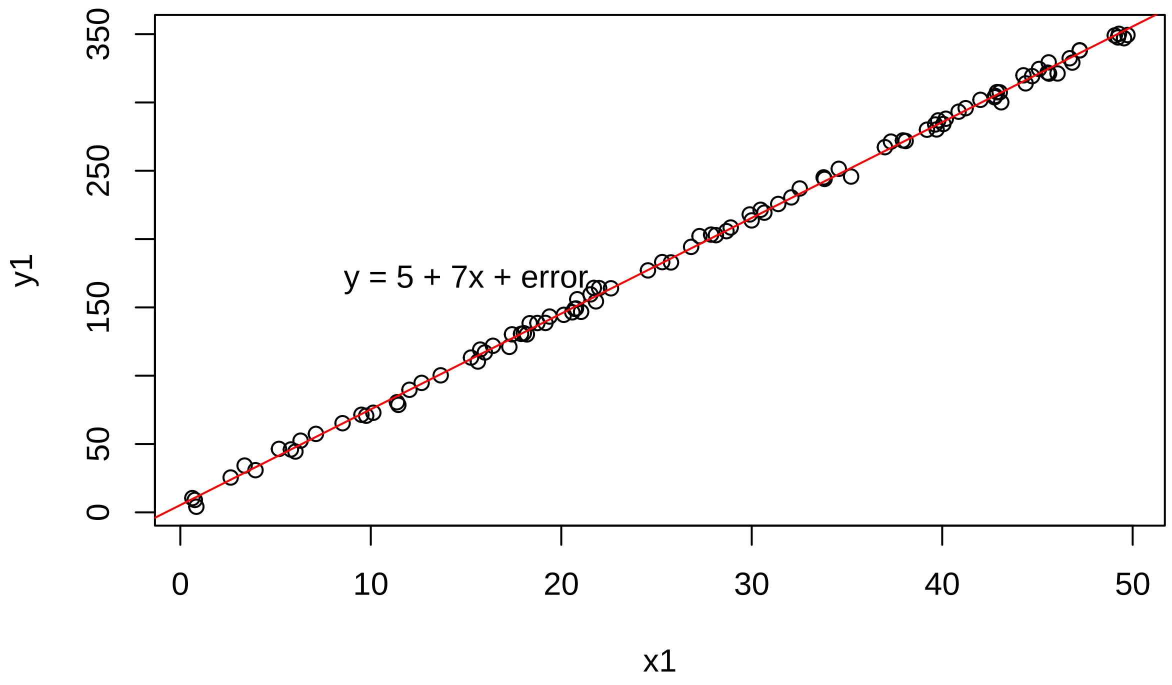
```
plot(x1,y1)
abline(lm_1,col ="red")
```



7.2c

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

```
plot(x1,y1)
abline(lm_1,col ="red")
text(x=15, y =170, "y = 5 + 7x + error")
```



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.

```r
x2 <-runif(100, 0, 50)
e2 <-rnorm(100, 0, 3)
y2<-10+1*x2+-0.1*x2^2+e2
stan_glm1<-rstanarm::stan_glm(y2~x2)
```

```
## 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 0.000129 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.29 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: 0.093 seconds (Warm-up)
## Chain 1:                0.108 seconds (Sampling)
## Chain 1:                0.201 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.2e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.22 seconds.
## Chain 2: Adjust your expectations accordingly!
## Chain 2:
## Chain 2:
## Chain 2: 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: 0.165 seconds (Warm-up)
## Chain 2:                0.101 seconds (Sampling)
## Chain 2:                0.266 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 2.2e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.22 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: 0.103 seconds (Warm-up)
## Chain 3:                0.116 seconds (Sampling)
## Chain 3:                0.219 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 2.8e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.28 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: 0.204 seconds (Warm-up)
## Chain 4:                0.079 seconds (Sampling)
## Chain 4:                0.283 seconds (Total)
## Chain 4:
```

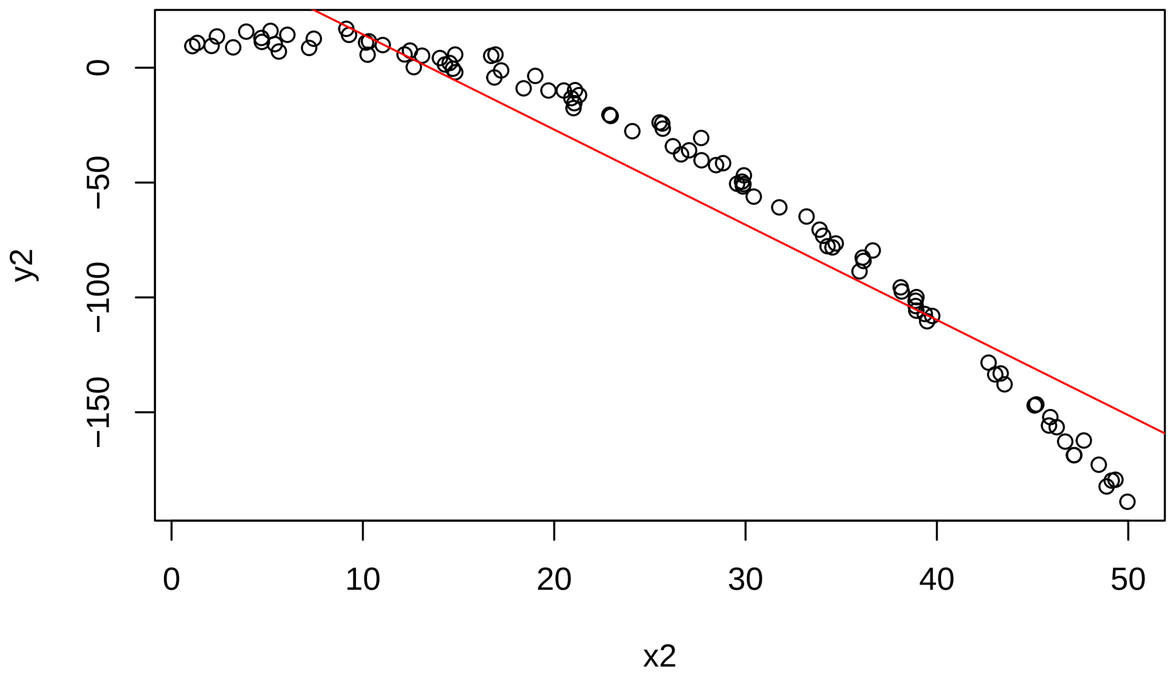
```r
summary(stan_glm1)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y2 ~ x2
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 100
##  predictors:   2
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) 55.9    3.9 50.8  56.0  61.0
## x2          -4.1    0.1 -4.3  -4.1  -4.0
## sigma       19.1    1.3 17.4  19.0  20.9
##
## Fit Diagnostics:
##            mean   sd    10%   50%   90%
## mean_PPD -51.7    2.7 -55.2 -51.7 -48.3
##
## The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for de
##
## MCMC diagnostics
##               mcse Rhat n_eff
## (Intercept)   0.1  1.0  3851
## x2            0.0  1.0  4096
## sigma         0.0  1.0  3779
## mean_PPD      0.0  1.0  3788
## log-posterior 0.0  1.0  1685
##
## 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? Best fit in this context refers to closest linear approximation of the model although that does not define the true shape of our data.

```r
plot(x2,y2)
abline(stan_glm1, col = "red")
```



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%.

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.

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.

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.

```
x3<-runif(100,0,20)
e3<-rnorm(100,0,5)
y3<- 2+3*x3+e3
lm_2<-lm(y3~x3)
stan_glm2<-rstanarm::stan_glm(y3~x3)
```

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

```
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 2.9e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.29 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: 0.089 seconds (Warm-up)
## Chain 1:                0.098 seconds (Sampling)
## Chain 1:                0.187 seconds (Total)
## Chain 1:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
## Chain 2:
## Chain 2: Gradient evaluation took 2.1e-05 seconds
## Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.21 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: 0.094 seconds (Warm-up)
## Chain 2:                0.073 seconds (Sampling)
## Chain 2:                0.167 seconds (Total)
## Chain 2:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 3).
## Chain 3:
## Chain 3: Gradient evaluation took 1.9e-05 seconds
## Chain 3: 1000 transitions using 10 leapfrog steps per transition would take 0.19 seconds.
## Chain 3: Adjust your expectations accordingly!
## Chain 3:
## Chain 3:
## Chain 3: 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: 0.084 seconds (Warm-up)
## Chain 3:                0.083 seconds (Sampling)
## Chain 3:                0.167 seconds (Total)
## Chain 3:
##
## SAMPLING FOR MODEL 'continuous' NOW (CHAIN 4).
## Chain 4:
## Chain 4: Gradient evaluation took 4e-05 seconds
## Chain 4: 1000 transitions using 10 leapfrog steps per transition would take 0.4 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: 0.117 seconds (Warm-up)
## Chain 4:                0.11 seconds (Sampling)
## Chain 4:                0.227 seconds (Total)
## Chain 4:
```

```
summary(lm_2)
```

```
##
## Call:
## lm(formula = y3 ~ x3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.903  -2.667  -0.098   2.683  12.095
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.74912    0.86073   2.032   0.0448 *
## x3           3.05322    0.08038  37.982   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.652 on 98 degrees of freedom
## Multiple R-squared:  0.9364, Adjusted R-squared:  0.9357
## F-statistic:  1443 on 1 and 98 DF,  p-value: < 2.2e-16
```

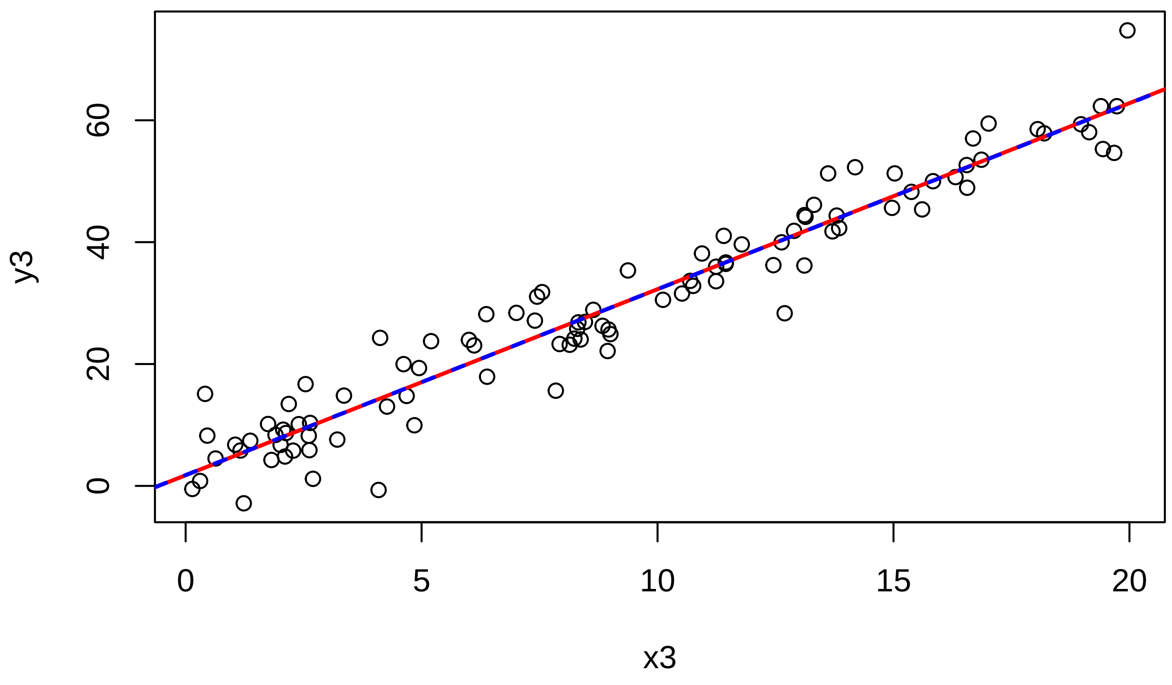
```
summary(stan_glm2)
```

```
##
## Model Info:
##  function:     stan_glm
##  family:       gaussian [identity]
##  formula:      y3 ~ x3
##  algorithm:    sampling
##  sample:       4000 (posterior sample size)
##  priors:       see help('prior_summary')
##  observations: 100
##  predictors:   2
##
## Estimates:
##               mean   sd   10%   50%   90%
## (Intercept) 1.8    0.9  0.6   1.8   2.9
## x3          3.1    0.1  2.9   3.1   3.2
## sigma       4.7    0.3  4.3   4.7   5.1
##
## Fit Diagnostics:
##            mean   sd   10%   50%   90%
## mean_PPD 29.3    0.7 28.4   29.3   30.1
##
## 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  3492
## x3            0.0  1.0  3378
## sigma         0.0  1.0  3309
## mean_PPD      0.0  1.0  3721
## log-posterior 0.0  1.0  1473
##
## For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample
```

8.8b

Plot the simulated data and the two fitted regression lines.

```r
plot(x3, y3)
abline(lm_2, col = "red", lwd = 2, lty = 1)
abline(a = coef(stan_glm2)[1], b = coef(stan_glm2)[2], col = "blue", lwd = 2, lty = 2)
```



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.

10.1 Regression with interactions:

Simulate 100 data points from the model, $y = b_0 + b_1 x + b_2 z + b_3 xz$ + error, with a continuous predictor x and a binary predictor z, coefficients $b = c(1, 2, -1, -2)$, and errors drawn independently from a normal distribution with mean 0 and standard deviation 3, as follows. For each data point i, first draw 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.

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.

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.

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.

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.

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.

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?

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?

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.

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.

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.

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.

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.

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.

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?

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.

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.

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.

10.8d

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

10.8e

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