Week 3 - Homework

STAT 420, Summer 2017, Dalpiaz

Exercise 1 (Using `lm` for Inference)

For this exercise we will use the `cats` dataset from the `MASS` package. You should use `?cats` to learn about the background of this dataset.

(a) Fit the following simple linear regression model in `R`. Use heart weight as the response and body weight as the predictor.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Store the results in a variable called `cat_model`. Use a t test to test the significance of the regression. Report the following:

- The null and alternative hypotheses
- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.01$
- A conclusion in the context of the problem

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

Solution:

```
library(MASS)
cat_model = lm(Hwt ~ Bwt, data = cats)
summary(cat_model)
```

```
##
## Call:
## lm(formula = Hwt ~ Bwt, data = cats)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5694 -0.9634 -0.0921  1.0426  5.1238
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.3567     0.6923  -0.515    0.607
## Bwt           4.0341     0.2503  16.119   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.452 on 142 degrees of freedom
## Multiple R-squared:  0.6466, Adjusted R-squared:  0.6441
## F-statistic: 259.8 on 1 and 142 DF,  p-value: < 2.2e-16
```

```
summary(cat_model)$coefficients["Bwt", "t value"] #test statistic
```

```
## [1] 16.11939
```

```r
summary(cat_model)$coefficients["Bwt", "Pr(>|t|)"] #p-value
```

```
## [1] 6.969045e-34
```

- $H_0 : \beta_1 = 0$
- $H_1 : \beta_1 \neq 0$
- Test statistic: $t = 16.1193908$
- P-value: $6.9690446 \times 10^{-34}$
- Decision: **Reject** H_0 at $\alpha = 0.01$.
- Conclusion: There is a linear relationship between heart weight and body weight.

(b) Calculate a 99% confidence interval for β_1. Give an interpretation of the interval in the context of the problem.

Solution:

```r
confint(cat_model, "Bwt", level = 0.99)
```

```
##        0.5 %   99.5 %
## Bwt 3.380656 4.687469
```

A 99% confidence interval for β_1 is given by

$$(3.3806562, 4.6874692).$$

Notice that this interval does **not** contain 0, which suggests that 0 is not a plausible value for β_1. This notion matches our result from the previous hypothesis test.

Interpretation: We are 99% confident that given a 1 kilogram increase in body weight, the average increase in heart weight will be between 3.3806562 and 4.6874692 grams.

(c) Calculate a 90% confidence interval for β_0. Give an interpretation of the interval in the context of the problem.

Solution:

```r
confint(cat_model, "(Intercept)", level = 0.90)
```

```
##                    5 %      95 %
## (Intercept) -1.502834 0.7895096
```

A 90% confidence interval for β_0 is given by

$$(-1.5028345, 0.7895096).$$

Interpretation: Mathematically, we are 90% confident that for a body weight of 0 kilograms, the average heart weight will be between -1.5028345 and 0.7895096 grams.

However, this confidence interval has no **practical** explanation because it is nonsense to consider the heart weight of a cat that weighs 0 kilograms.

(d) Use a 95% confidence interval to estimate the mean heart weight for body weights of 2.5 and 3.0 kilograms Which of the two intervals is wider? Why?

Solution:

```
new_body_weights = data.frame(Bwt = c(2.5, 3.0))
(heart_weight_ci = predict(cat_model, newdata = new_body_weights,
                           interval = c("confidence"), level = 0.95))
```

```
##         fit       lwr       upr
## 1  9.728494  9.464902  9.992087
## 2 11.745526 11.469954 12.021097
```

We are 95% confident that the mean heart weight for a body weight of 2.5 kilograms is in the interval

$$(9.4649016, 9.992087).$$

We are 95% confident that the mean heart weight for a body weight of 3.0 kilograms is in the interval

$$(11.469954, 12.0210973).$$

```
mean(cats$Bwt)
```

```
## [1] 2.723611
```

```
range(cats$Bwt)
```

```
## [1] 2.0 3.9
```

The interval for a body weight of 3.0 kilograms is larger since it is further from the sample mean body weight. Also, note that both body weights fall within the range of observed body weights.

```
heart_weight_ci = unname(heart_weight_ci) #removes name information for future display
heart_weight_ci[, 2:3]
```

```
##           [,1]      [,2]
## [1,]  9.464902  9.992087
## [2,] 11.469954 12.021097
```

```
diff(heart_weight_ci[1, 2:3])
```

```
## [1] 0.5271854
```

```
diff(heart_weight_ci[2, 2:3])
```

```
## [1] 0.5511433
```

```
diff(heart_weight_ci[1, 2:3]) < diff(heart_weight_ci[2, 2:3])
```

```
## [1] TRUE
```

(e) Use a 95% prediction interval to predict the heart weight for body weights of 2.5 and 4.0 kilograms.

Solution:

```
new_body_weights = data.frame(Bwt = c(2.5, 4.0))
(heart_weight_pi = predict(cat_model, newdata = new_body_weights,
                           interval = c("prediction"), level = 0.95))
```

```
##         fit       lwr      upr
## 1  9.728494  6.845352 12.61164
## 2 15.779588 12.830180 18.72900
```

We are 95% confident that a *new* **observation** of heart weight for a body weight of 2.5 kilograms is in the interval

$$(6.8453519, 12.6116367).$$

We are 95% confident that a *new* **observation** of heart weight for a body weight of 4.0 kilograms is in the interval

$$(12.8301804, 18.7289963).$$

Note that the prediction interval for a body weight of 2.5 kilograms is wider than the confidence interval for the same body weight found in **(d)**.

```
heart_weight_pi = unname(heart_weight_pi)
diff(heart_weight_ci[1, 2:3]) < diff(heart_weight_pi[1, 2:3])
```

```
## [1] TRUE
```

(f) Create a scatterplot of the data. Add the regression line, 95% confidence bands, and 95% prediction bands.

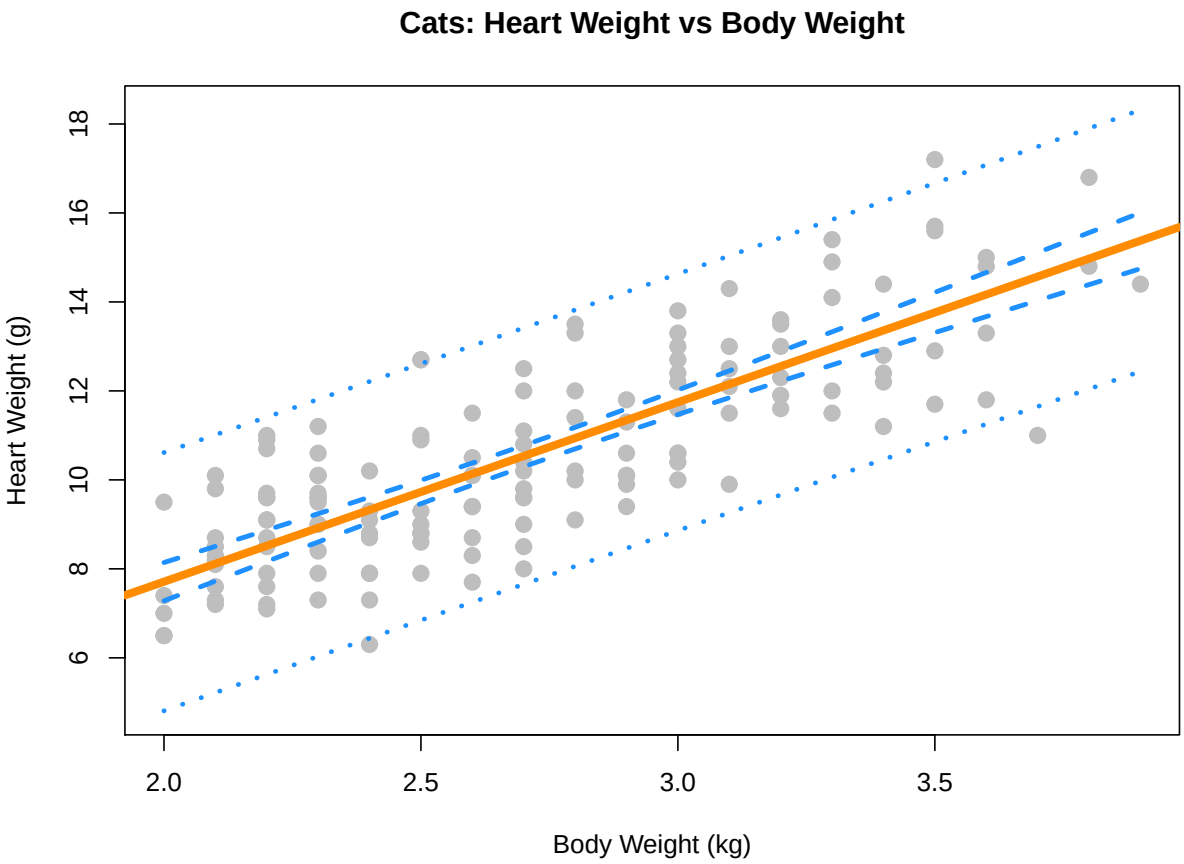
Solution:

```
bw_grid = seq(min(cats$Bwt), max(cats$Bwt), by = 0.01)

hw_ci_band = predict(cat_model, newdata = data.frame(Bwt = bw_grid),
                     interval = "confidence")
hw_pi_band = predict(cat_model, newdata = data.frame(Bwt = bw_grid),
                     interval = "prediction")

plot(Hwt ~ Bwt, data = cats,
     xlab = "Body Weight (kg)",
     ylab = "Heart Weight (g)",
     main = "Cats: Heart Weight vs Body Weight",
     pch  = 20,
     cex  = 2,
     col  = "grey",
     ylim = c(min(hw_pi_band), max(hw_pi_band)))

abline(cat_model, lwd = 5, col = "darkorange")
lines(bw_grid, hw_ci_band[,"lwr"], col = "dodgerblue", lwd = 3, lty = 2)
lines(bw_grid, hw_ci_band[,"upr"], col = "dodgerblue", lwd = 3, lty = 2)
lines(bw_grid, hw_pi_band[,"lwr"], col = "dodgerblue", lwd = 3, lty = 3)
lines(bw_grid, hw_pi_band[,"upr"], col = "dodgerblue", lwd = 3, lty = 3)
```



Notice that, while the vast majority of the data points are within the prediction bands, very few points are within the confidence bands.

Exercise 2 (Using `lm` for Inference)

For this exercise we will use the `diabetes` dataset, which can be found in the `faraway` package.

(a) Fit the following simple linear regression model in `R`. Use the total cholesterol as the response and weight as the predictor.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Store the results in a variable called `cholesterol_model`. Use a t test to test the significance of the regression. Report the following:

- The null and alternative hypotheses
- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.05$
- A conclusion in the context of the problem

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

Solution:

```
library(faraway)
cholesterol_model = lm(chol ~ weight, data = diabetes)
summary(cholesterol_model)$coefficients["weight", "t value"] #test statistic
```

```
## [1] 1.339096
```

```
summary(cholesterol_model)$coefficients["weight", "Pr(>|t|)"] #p-value
```

```
## [1] 0.1813018
```

- $H_0 : \beta_1 = 0$, $Y_i = \beta_0 + \epsilon_i$
- $H_1 : \beta_1 \neq 0$, $Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$
- Test statistic: $t = 1.3390958$
- P-value: 0.1813018
- Decision: **Fail to Reject** H_0 at $\alpha = 0.05$.
- Conclusion: There is **not** a linear relationship between cholesterol and weight.

(b) Fit the following simple linear regression model in `R`. Use HDL as the response and weight as the predictor.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Store the results in a variable called `hdl_model`. Use a t test to test the significance of the regression. Report the following:

- The null and alternative hypotheses
- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.05$
- A conclusion in the context of the problem

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

Solution:

```
hdl_model = lm(hdl ~ weight, data = diabetes)
summary(hdl_model)$coefficients["weight", "t value"] #test statistic
```

```
## [1] -6.075257
```

```
summary(hdl_model)$coefficients["weight", "Pr(>|t|)"] #p-value
```

```
## [1] 2.890526e-09
```

- $H_0 : \beta_1 = 0$, $Y_i = \beta_0 + \epsilon_i$
- $H_1 : \beta_1 \neq 0$, $Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$
- Test statistic: $t = -6.0752572$
- P-value: 2.8905259×10^{-9}
- Decision: **Reject** H_0 at $\alpha = 0.05$.
- Conclusion: There is a linear relationship between HDL and weight.

Exercise 3 (Inference "without" `lm`)

Write a function named `get_p_val_beta_1` that performs the test

$$H_0 : \beta_1 = \beta_{10} \quad \text{vs} \quad H_1 : \beta_1 \neq \beta_{10}$$

for the linear model

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i.$$

The function should take two inputs:

- A model object that is the result of fitting the SLR model with `lm()`
- A hypothesized value of β_1, β_{10}, with a default value of 0

The function should return a named vector with elements:

- `t`, which stores the value of the test statistic for performing the test
- `p_val`, which stores the p-value for performing the test

(a) After writing the function, run these three lines of code:

```
get_p_val_beta_1(cat_model, beta_1 = 4.2)
get_p_val_beta_1(cholesterol_model)
get_p_val_beta_1(hdl_model)
```

Solution:

```
get_p_val_beta_1 = function(model, beta_1 = 0) {
  n = length(resid(model))
  est = summary(model)$coefficients[2, "Estimate"]
  se = summary(model)$coefficients[2, "Std. Error"]
  t = (est - beta_1) / se
  p_val = 2 * pt(abs(t), df = n - 2, lower.tail = FALSE)
  c(t = t, p_val = p_val)
}
```

```
get_p_val_beta_1(cat_model, beta_1 = 4.2)
```

```
##          t      p_val
## -0.6630557  0.5083698
```

```
get_p_val_beta_1(cholesterol_model)
```

```
##         t     p_val
## 1.3390958 0.1813018
```

```
get_p_val_beta_1(hdl_model)
```

```
##             t         p_val
## -6.075257e+00  2.890526e-09
```

(b) Return to the goalies dataset from the previous homework, which is stored in `goalies.csv`. Fit a simple linear regression model with `W` as the response and `MIN` as the predictor. Store the results in a variable called `goalies_model_min`. After doing so, run these three lines of code:

```
get_p_val_beta_1(goalies_model_min)
get_p_val_beta_1(goalies_model_min, beta_1 = coef(goalies_model_min)[2])
get_p_val_beta_1(goalies_model_min, beta_1 = 0.008)
```

Solution:

```
goalies = read.csv("goalies.csv")
goalies_model_min = lm(W ~ MIN, data = goalies)
```

```
get_p_val_beta_1(goalies_model_min)
```

```
##       t   p_val
## 154.724   0.000
```

```
get_p_val_beta_1(goalies_model_min, beta_1 = coef(goalies_model_min)[2])
```

```
##     t.MIN p_val.MIN
##         0         1
```

```
get_p_val_beta_1(goalies_model_min, beta_1 = 0.008)
```

```
##            t        p_val
## -3.036956036  0.002477199
```

Exercise 4 (Simulating Sampling Distributions)

For this exercise we will simulate data from the following model:

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Where $\epsilon_i \sim N(0, \sigma^2)$. Also, the parameters are known to be:

- $\beta_0 = 3$
- $\beta_1 = 0.75$
- $\sigma^2 = 25$

We will use samples of size $n = 42$.

(a) Simulate this model 1500 times. Each time use `lm()` to fit a simple linear regression model, then store the value of $\hat{\beta}_0$ and $\hat{\beta}_1$. Set a seed using **your** birthday before performing the simulation. Note, we are simulating the x values once, and then they remain fixed for the remainder of the exercise.

```
birthday = 18760613
set.seed(birthday)
n = 42
x = seq(0, 20, length = n)
```

Solution:

```
beta_0    = 3
beta_1    = 0.75
sigma     = 5
true_line = beta_0 + beta_1 * x

num_sim   = 1500
beta_hats = matrix(0, num_sim, 2)
for (i in 1:num_sim) {
  y = true_line + rnorm(n, mean = 0, sd = sigma)
  beta_hats[i, ] = coef(lm(y ~ x))
}
```

```
beta_0_hats = beta_hats[, 1]
beta_1_hats = beta_hats[, 2]
```

(b) For the *known* values of x, what is the expected value of $\hat{\beta}_1$?

Solution:

$$\hat{\beta}_1 \sim N\left(\beta_1, \sigma^2 / S_{xx}\right)$$

$$\mathrm{E}[\hat{\beta}_1] = 0.75$$

(c) For the known values of x, what is the standard deviation of $\hat{\beta}_1$?

Solution:

```
Sxx = sum((x - mean(x)) ^ 2)
sigma / sqrt(Sxx)
```

```
## [1] 0.1304859
```

$$\mathrm{SD}[\hat{\beta}_1] = 0.1304859$$

(d) What is the mean of your simulated values of $\hat{\beta}_1$? Does this make sense given your answer in **(b)**?

Solution:

```
mean(beta_1_hats)
```

```
## [1] 0.7487103
```

Yes, this is close to the true mean of β_1.

(e) What is the standard deviation of your simulated values of $\hat{\beta}_1$? Does this make sense given your answer in **(c)**?

Solution:

```
sd(beta_1_hats)
```

```
## [1] 0.1329698
```

Yes, this is close to the true standard deviation of β_1.

(f) For the known values of x, what is the expected value of $\hat{\beta}_0$?

Solution:

$$\hat{\beta}_0 \sim N\left(\beta_0, \sigma^2(1/n + \bar{x}^2/S_{xx})\right)$$

$$\mathrm{E}[\hat{\beta}_0] = 3$$

(g) For the known values of x, what is the standard deviation of $\hat{\beta}_0$?

Solution:

```
sigma * sqrt(1 / n + mean(x) ^ 2 / Sxx)
```

```
## [1] 1.515881
```

$$\text{SD}[\hat{\beta}_0] = 1.5158812$$

(h) What is the mean of your simulated values of $\hat{\beta}_0$? Does this make sense given your answer in **(f)**?

Solution:

```
mean(beta_0_hats)
```

```
## [1] 2.993735
```

Yes, this is close to the true mean of β_0.

(i) What is the standard deviation of your simulated values of $\hat{\beta}_0$? Does this make sense given your answer in **(g)**?

Solution:

```
sd(beta_0_hats)
```

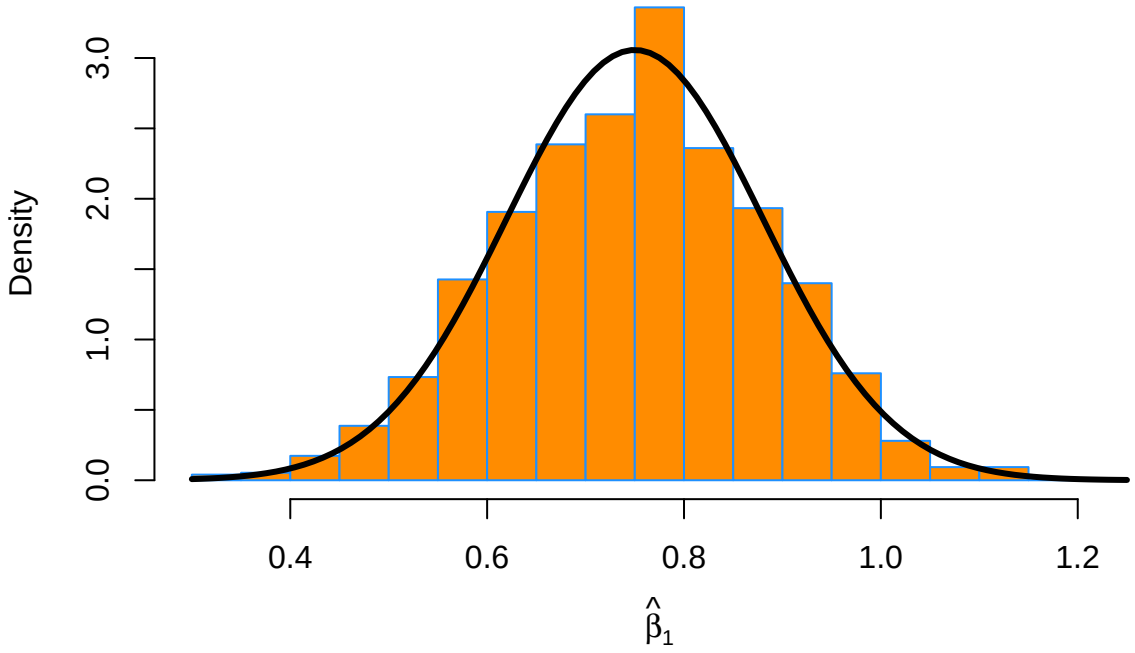
```
## [1] 1.567255
```

Yes, this is close to the true standard deviation of β_0.

(j) Plot a histogram of your simulated values for $\hat{\beta}_1$. Add the normal curve for the true sampling distribution of $\hat{\beta}_1$.

Solution:

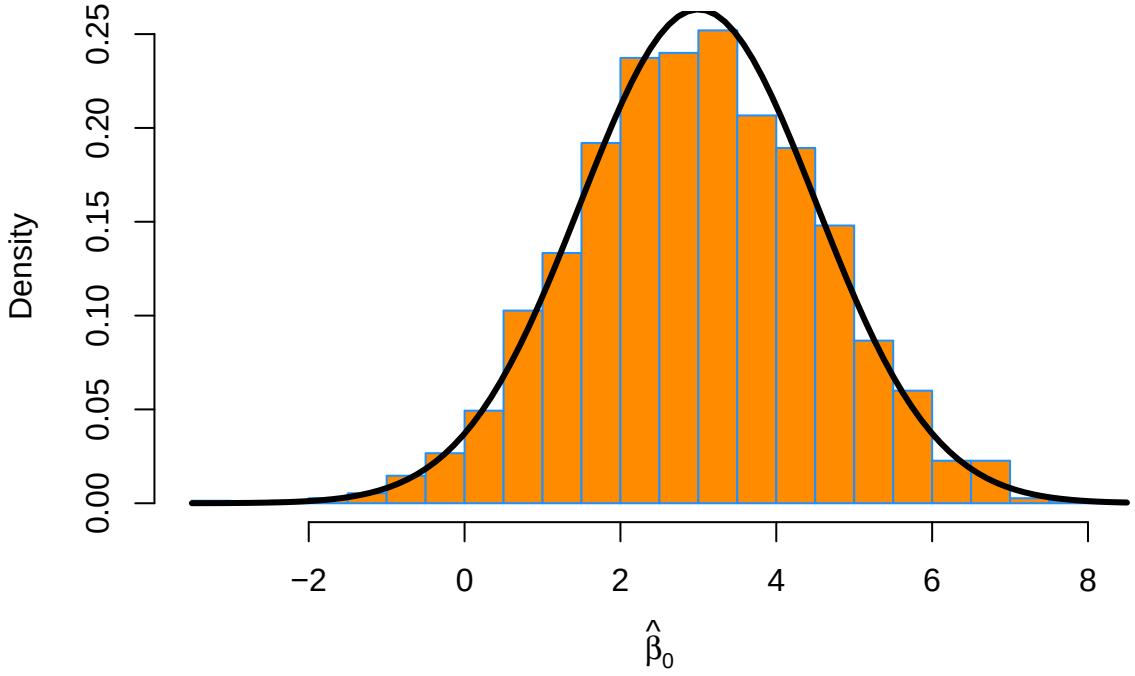
```
hist(beta_1_hats, breaks = 25, col = "darkorange", border = "dodgerblue",
     prob = TRUE, xlab = expression(hat(beta)[1]), main = "")
e_beta_1_hat  = beta_1
sd_beta_1_hat = sigma / sqrt(Sxx)
curve(dnorm(x, mean = e_beta_1_hat, sd = sd_beta_1_hat), add = TRUE, lwd = 3)
```



(k) Plot a histogram of your simulated values for $\hat{\beta}_0$. Add the normal curve for the true sampling distribution of $\hat{\beta}_0$.

Solution:

```
hist(beta_0_hats, breaks = 25, col = "darkorange", border = "dodgerblue",
     prob = TRUE, xlab = expression(hat(beta)[0]), main = "")
e_beta_0_hat  = beta_0
sd_beta_0_hat = sigma * sqrt(1 / n + mean(x) ^ 2 / Sxx)
curve(dnorm(x, mean = e_beta_0_hat, sd = sd_beta_0_hat), add = TRUE, lwd = 3)
```



Exercise 5 (Simulating Confidence Intervals)

For this exercise we will simulate data from the following model:

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Where $\epsilon_i \sim N(0, \sigma^2)$. Also, the parameters are known to be:

- $\beta_0 = 1$
- $\beta_1 = 3$
- $\sigma^2 = 16$

We will use samples of size $n = 20$.

Our goal here is to use simulation to verify that the confidence intervals really do have their stated confidence level. Do **not** use the `confint()` function for this entire exercise.

(a) Simulate this model 2000 times. Each time use `lm()` to fit a simple linear regression model, then store the value of $\hat{\beta}_0$ and s_e. Set a seed using **your** birthday before performing the simulation. Note, we are simulating the x values once, and then they remain fixed for the remainder of the exercise.

```
birthday = 18760613
set.seed(birthday)
n = 20
x = seq(-5, 5, length = n)
```

Solution:

```
beta_0    = 1
beta_1    = 3
sigma     = 4
true_line = beta_0 + beta_1 * x

num_sim    = 2000
beta_hat_0 = rep(0, num_sim)
s_e        = rep(0, num_sim)

for (i in 1:num_sim) {
  y             = true_line + rnorm(n, 0, sigma)
  beta_hat_0[i] = coef(lm(y ~ x))[1]
  s_e[i]        = summary((lm(y ~ x)))$sigma
}
```

(b) For each of the $\hat{\beta}_0$ that you simulated, calculate a 90% confidence interval. Store the lower limits in a vector `lower_90` and the upper limits in a vector `upper_90`. Some hints:

- You will need to use `qt()` to calculate the critical value, which will be the same for each interval.
- Remember that `x` is fixed, so S_{xx} will be the same for each interval.
- You could, but do not need to write a `for` loop. Remember vectorized operations.

Solution:

Recall,

$$\hat{\beta}_0 \pm t_{\alpha/2, n-2} \cdot s_e \sqrt{\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}}}$$

```
alpha = 0.10
t_crit_90 = -qt(alpha / 2, df = n - 2)
Sxx = sum((x - mean(x)) ^ 2)

lower_90 = beta_hat_0 - t_crit_90 * s_e * sqrt(1 / n + mean(x) ^ 2 / Sxx)
upper_90 = beta_hat_0 + t_crit_90 * s_e * sqrt(1 / n + mean(x) ^ 2 / Sxx)
```

(c) What proportion of these intervals contain the true value of β_0?

Solution:

```
mean(lower_90 < 1 & 1 < upper_90)
```

```
## [1] 0.8895
```

Unsurprisingly, the result is near 90%.

(d) Based on these intervals, what proportion of the simulations would reject the test $H_0 : \beta_0 = 0$ vs $H_1 : \beta_0 \neq 0$ at $\alpha = 0.10$?

Solution:

```
1 - mean(lower_90 < 0 & 0 < upper_90)
```

```
## [1] 0.279
```

(e) For each of the $\hat{\beta}_0$ that you simulated, calculate a 99% confidence interval. Store the lower limits in a vector `lower_99` and the upper limits in a vector `upper_99`.

Solution:

Recall,

$$\hat{\beta}_0 \pm t_{\alpha/2, n-2} \cdot s_e \sqrt{\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}}}$$

```
alpha = 0.01
t_crit_99 = -qt(alpha / 2, df = n - 2)
Sxx = sum((x - mean(x)) ^ 2)

lower_99 = beta_hat_0 - t_crit_99 * s_e * sqrt(1 / n + mean(x) ^ 2 / Sxx)
upper_99 = beta_hat_0 + t_crit_99 * s_e * sqrt(1 / n + mean(x) ^ 2 / Sxx)
```

Note that we could have stored confidence intervals directly when performing the simulation, using `confint()`. However, then we would not have been so easily able to modify the confidence level. We would have needed to perform the simulation again.

(f) What proportion of these intervals contain the true value of β_0?

Solution:

```
mean(lower_99 < 1 & 1 < upper_99)
```

```
## [1] 0.9865
```

Unsurprisingly, the result is near 99%.

(g) Based on these intervals, what proportion of the simulations would reject the test $H_0 : \beta_0 = 0$ vs $H_1 : \beta_0 \neq 0$ at $\alpha = 0.01$?

Solution:

```
1 - mean(lower_99 < 0 & 0 < upper_99)
```

```
## [1] 0.064
```