Ch. 4 Extra Exercises: Part 2

March 4, 2019

1. On pg. 90 of CH 4 in *Statistical Rethinking*, the book uses a function `cov2cor()` to convert a covariance matrix into a correlation matrix. Write your own function which does the same and write out the computations using matrix algebra notation.

Solution:

```r
my_cov2cor <- function(cov_mat){
  diag <- diag(cov_mat)
  diag_inv_sqrt <- (diag)^(-1/2)
  corr_mat <- diag_inv_sqrt * cov_mat * rep(diag_inv_sqrt, each = dim(cov_mat)[1L])

  return(corr_mat)
}
```

Now let's compare the output of this function to the built-in `cov2cor()` function:

```
#Test run using data from Ch. 4 pg. 87-90
data(Howell1)
d <- Howell1
d2 <- d[d$age >= 18,]

flist <- alist(
  height ~ dnorm(mu, sigma),
  mu ~ dnorm(178, 20),
  sigma ~ dunif(0, 50)
)

m4.1 <- map(flist, data=d2)

cov2cor(vcov(m4.1))

##                mu       sigma
## mu    1.000000000 0.002073111
## sigma 0.002073111 1.000000000

my_cov2cor(vcov(m4.1))

##                mu       sigma
## mu    1.000000000 0.002073111
## sigma 0.002073111 1.000000000
```

2. On pg. 92 of CH 4 in *Statistical Rethinking*, the author talks about regression to the mean and shrinkage. Read more about shrinkage estimation (including those articles I gave you last week), to explain more about the purpose and benefit of using shrinkage estimation. Include citations for any references you use.

Solution:

Shrinkage estimation generallly reduces the variance in parameter estimation. This may serve to improve out-of-sample prediction by reducing the variance of the model while increasing the bias. If the ”correct” parameters for a particular shrinkage method are chosen, then the trade-off between variance and bias should result in a model with better predictive capability when compared with an OLS model.

Ridge regression, one such shrinkage method, is very similar to OLS regression. It includes a penalty term with the RSS calculation which penalizes the model for having large β values. The parameter $\lambda \geq 0$ controls the size of the penalty, with $\lambda = 0$ simplifying the model to OLS regression. The penalty term shrinks the β terms towards 0 (but never reaching 0).

Lasso regression is another shrinkage method which decreases the variance of the β values. Lasso minimizes the RSS, but includes the constraint that $\sum_{j=1}^{p} |\beta_j| \leq t$. If t is chosen sufficiently small, then some of the β values will be reduced all the way to 0. In this manner, lasso regression is able to perform a sort of continuous subset selection.

Cite later:

- Introduction to Statistical Learning
- Elements of Statistical Learning
- Regression Modeling Strategies
- Brown (2007)
- Copas (1983)

3. On pg. 99 of CH 4 in *Statistical Rethinking*, the author writes, "But in more complex models, strong [parameter] correlations like this can make it difficult to fit the model to the data."

   (a) Explain why correlations between pairs of parameters is a problem.

   (b) Why does the author only center the x-variable, `weight`, and not the y-variable `height`?

Solution:

(a)

(b) Centering the x-variable but not the y-variable retains the estimate and interpretability of the β coefficient and improves the interpretability of the intercept α. The intercept is now interpreted as the value of $\hat{y}$ when x is equal to the mean of the data. This ensures that the intercept is of practical importance, as it does not make sense to consider the case where $x = 0$ in many regression equations.

4. Use the `d2` data from the chapter to answer the following questions, which are extensions of **4H2**:

   (a) Fit a frequentist simple linear regression with `weight` as the explanatory variable and `height` as the response variable. Make a scatterplot of the data and plot both the frequentist and Bayesian lines. Is there much of a difference?

(b) Check the regression assumptions on your frequentist model. Include any relevant tables/graphs with your assumption checks.

(c) Are correlations between pairs of parameters a problem in frequentist regression? Calculate the variance-covariance matrix for $\hat{\boldsymbol{\beta}} = (\hat{\beta}_0, \hat{\beta}_1)$ using the formulas in the appendix to CH 3 of *Regression Analysis by Example* (your regression textbook).

(d) Does centering `weight` make a difference in the correlations between parameters in frequentist regression? Does it make a difference in the parameter estimates themselves? Try using the function `scale()` to center `weight` in your code.

(e) Use the function `confint()` to create a 95% confidence interval of the slope for your frequentist regression. Calculate a 95% posterior probability interval for the slope for your Bayesian regression. Interpret both and explain the difference between the two.

(f) Create a figure with 4 scatterplots, each with `weight` on the x-axis and `height` on the y-axis. On the first plot, add the Bayesian fitted model and add the 95% HPDI intervals (like Fig. 4.8). On the second plot, add the Bayesian fitted model and add the 95% PI intervals (also like Fig. 4.8). On the third plot, add the frequentist fitted model and the 95% confidence intervals for prediction (like slide 35 in Lecture 3 of the regression class). Finally, on the fourth plot, add the frequentist fitted model and the 95% prediction intervals for prediction (also like slide 35 in Lecture 3 of the regression class). Interpret all four plots and explain the connections between them.

Solution:

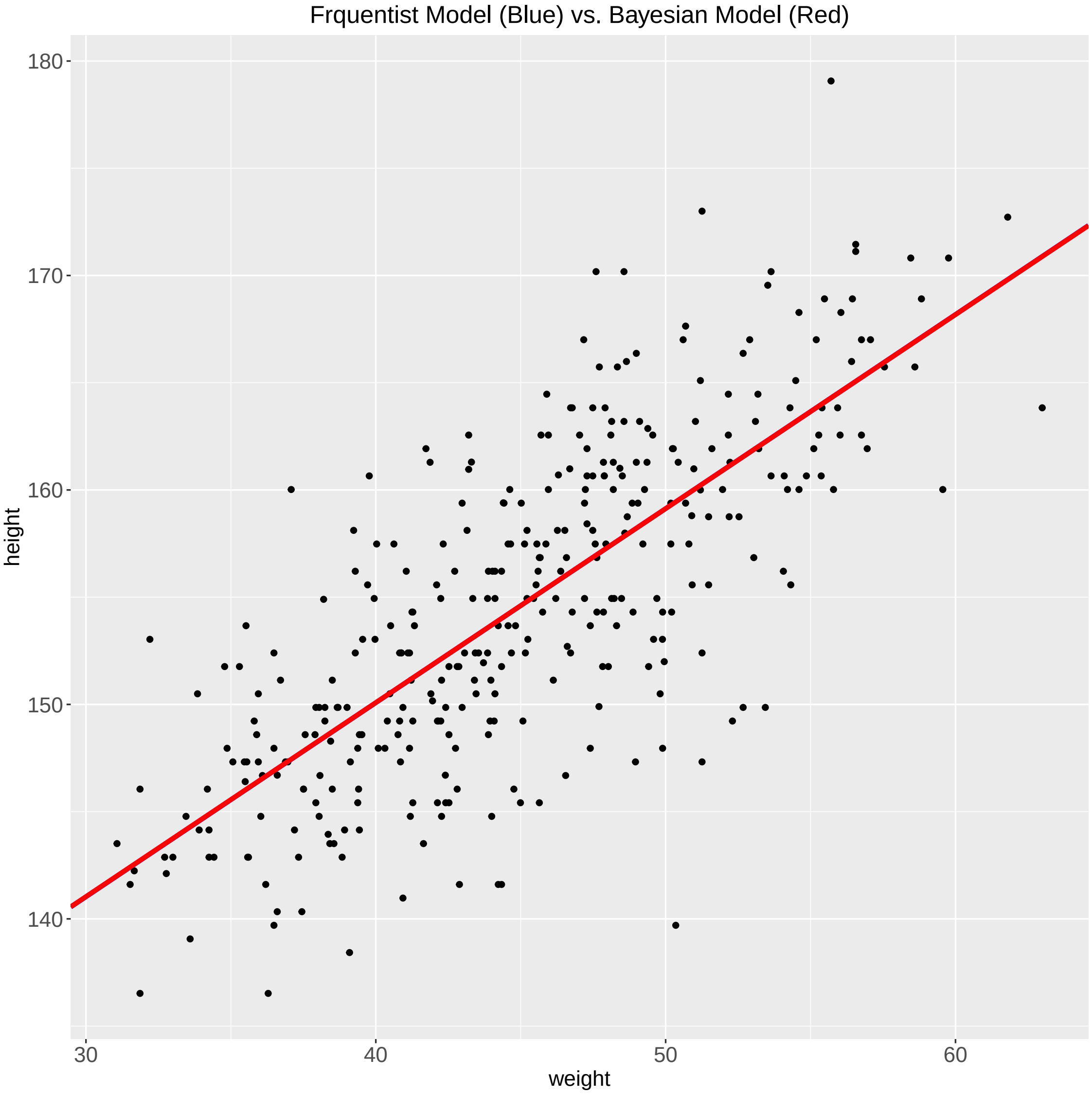
(a)

```
lm.1 <- lm(height ~ weight, data = d2)

summary(lm.1)

##
## Call:
## lm(formula = height ~ weight, data = d2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.7464  -2.8835   0.0222   3.1424  14.7744
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 113.87939    1.91107   59.59   <2e-16 ***
## weight        0.90503    0.04205   21.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.086 on 350 degrees of freedom
## Multiple R-squared:  0.5696,Adjusted R-squared:  0.5684
## F-statistic: 463.3 on 1 and 350 DF,  p-value: < 2.2e-16
```

```
m4.3 <- map(
  alist(
    height ~ dnorm(mu, sigma),
    mu <- a + b*weight,
    a ~ dnorm(156, 100),
    b ~ dnorm(0, 10),
    sigma ~ dunif(0, 50)
  ),
data = d2)

precis(m4.3)
```

```
##         Mean StdDev   5.5%  94.5%
## a     113.90   1.91 110.85 116.94
## b       0.90   0.04   0.84   0.97
## sigma   5.07   0.19   4.77   5.38
```

```
ggplot(data = d2,mapping = aes(x = weight,y = height)) +
  geom_point() +
  geom_abline(color = "blue", slope = lm.1$coefficients[2],
              intercept = lm.1$coefficients, size = 1.5,
              show.legend = TRUE) +
  geom_abline(color = "red", slope = coef(m4.3)["b"],
              intercept = coef(m4.3)["a"], size = 1.5,
              show.legend = TRUE) +
  ggtitle("Frquentist Model (Blue) vs. Bayesian Model (Red)") + theme.info
```



The two regression lines are nearly identical.

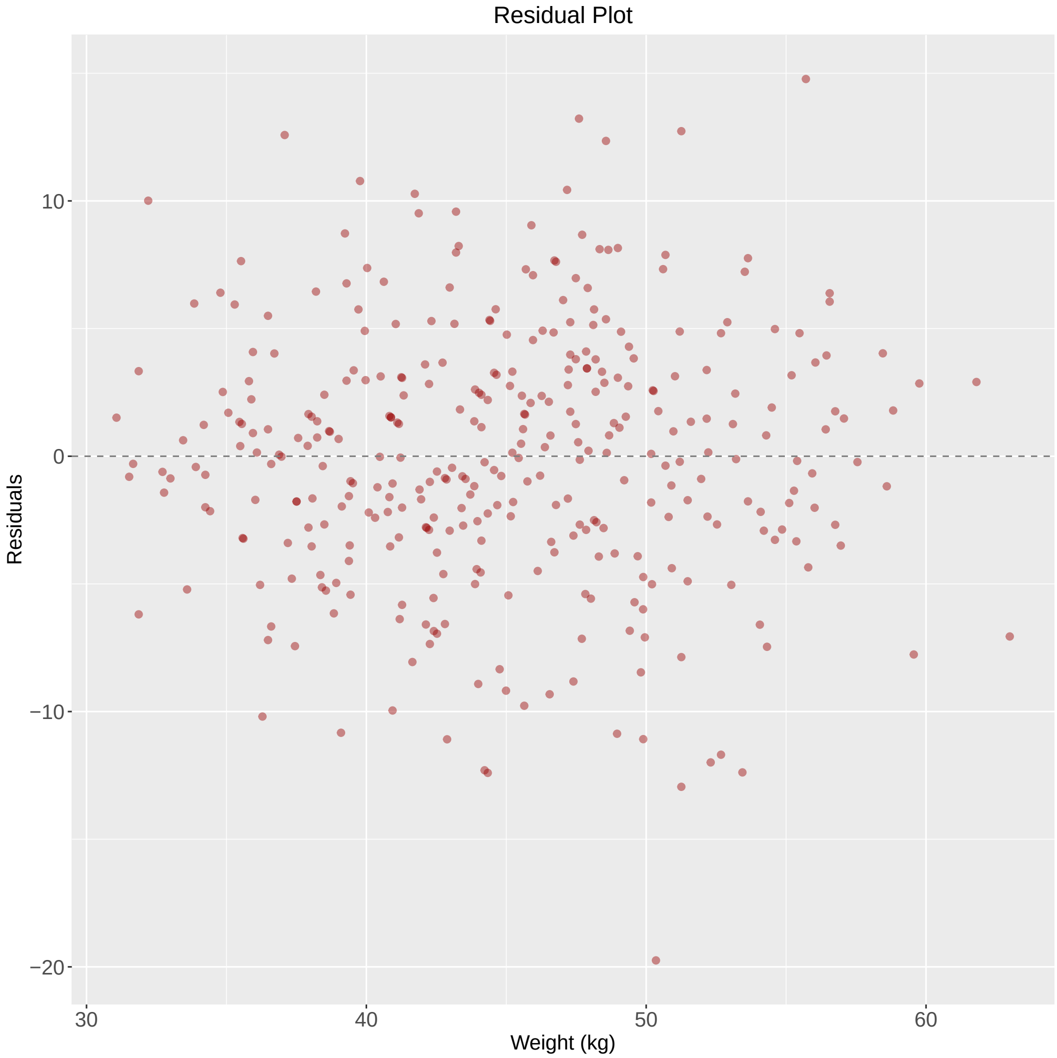
(b) Regression assumptions:

1. Fixed x with no measurement error:
   The x values are not fixed because we observed each weight value; they were not assigned before the regression. They are also not measured without error as any scale will likely introduce a small amount of measurement noise. However, on a well-calibrated scale, the amount of measurement noice introduced is likely ignorable.

2. x and y linearly related:
   We can see from the plot above that x and y are clearly linearly related.

3. Constant variance:

```
y.augment <- augment(lm.1)
y.augment

## # A tibble: 352 x 10
##    .rownames height weight .fitted .se.fit .resid    .hat .sigma .cooksd
##  * <chr>      <dbl>  <dbl>   <dbl>   <dbl>  <dbl>   <dbl>  <dbl>   <dbl>
##  1 1           152.   47.8    157.   0.296 -5.40  0.00339   5.09 1.92e-3
##  2 2           140.   36.5    147.   0.449 -7.20  0.00778   5.08 7.92e-3
##  3 3           137.   31.9    143.   0.615 -6.19  0.0146    5.08 1.12e-2
##  4 4           157.   53.0    162.   0.434 -5.04  0.00727   5.09 3.62e-3
##  5 5           145.   41.3    151.   0.313 -5.82  0.00378   5.08 2.50e-3
##  6 6           164.   63.0    171.   0.804 -7.06  0.0250    5.08 2.53e-2
##  7 7           149.   38.2    148.   0.392  0.734 0.00595   5.09 6.27e-5
##  8 8           169.   55.5    164.   0.518  4.82  0.0104    5.09 4.75e-3
##  9 9           148.   34.9    145.   0.505  2.52  0.00984   5.09 1.23e-3
## 10 10          165.   54.5    163.   0.483  1.91  0.00900   5.09 6.45e-4
## # ... with 342 more rows, and 1 more variable: .std.resid <dbl>

y.augment %>% ggplot(aes(x=weight, y=.resid)) +
  geom_point(color="#99000070", size=2) +
  ggtitle("Residual Plot") +
  labs(x="Weight (kg)", y="Residuals") +
  geom_hline(yintercept=0, color="gray50",  lty=2) +
  theme.info
```



We can see from the plot above that the variance in the residuals is constant regardless of the individual's weight.

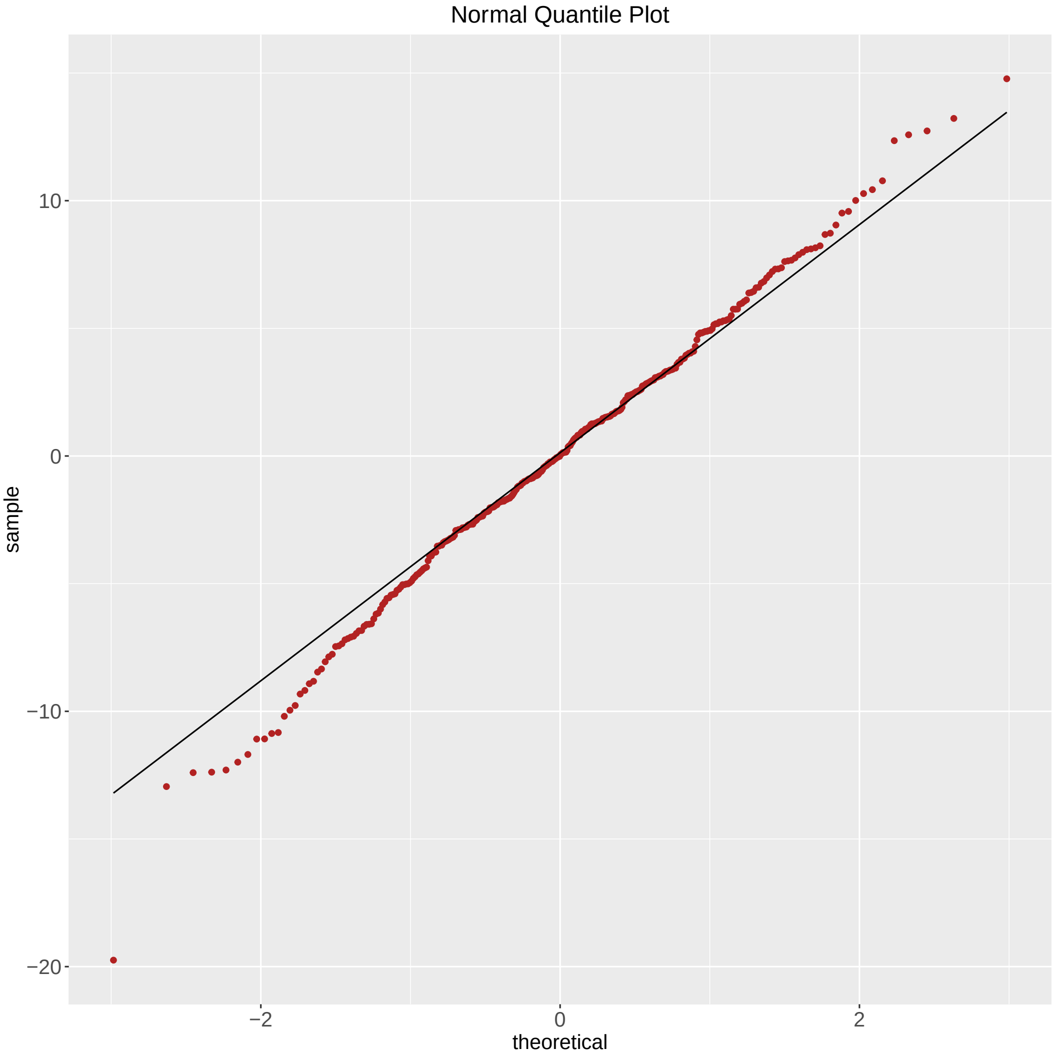
4. Mean of the residuals = 0:

```
mean(y.augment$.resid)

## [1] 5.337416e-17
```

As you can see, the mean of the residuals is very close to 0.

5. Normally distributed errors:

```
y.augment %>%  ggplot(aes(sample=.resid)) +
  stat_qq(col = "firebrick") +
  stat_qq_line() +
  ggtitle("Normal Quantile Plot") +
  theme.info
```



The Normal Quantile Plot shows a rougly straight line. There is some deviation towards each end of the line, indicating that the data comes from a heavy-tailed distribution. However, the deviation is small enough that we may still be able to run our regression with minimal error.

(c)

```
calculate_variance_covariance_matrix <- function(x_mat, y_mat){
  x_var <- as.matrix(x_mat)
  intercept <- matrix(rep(1, nrow(x_var)), ncol = 1)

  X <- cbind(intercept, x_var)

  Y <- as.matrix(y_mat)

  C <- solve(t(X) %*% X)

  beta_hat <- C %*% t(X) %*% Y

  var_covar <- C * var(Y)[1]

  return(var_covar)
}

print(calculate_variance_covariance_matrix(d2$weight, d2$height))
```

```
##             [,1]         [,2]
## [1,]  8.462223 -0.184304039
## [2,] -0.184304  0.004096511
```

(d)

```
d2$centered_weight <- scale(d2$weight, scale = FALSE)

lm.2 <- lm(height ~ centered_weight, data = d2)

summary(lm.2)
```

```
##
## Call:
## lm(formula = height ~ centered_weight, data = d2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -19.7464  -2.8835   0.0222   3.1424  14.7744
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     154.59709    0.27110  570.25   <2e-16 ***
## centered_weight   0.90503    0.04205   21.52   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.086 on 350 degrees of freedom
## Multiple R-squared:  0.5696,Adjusted R-squared:  0.5684
## F-statistic: 463.3 on 1 and 350 DF,  p-value: < 2.2e-16
```

```
print(calculate_variance_covariance_matrix(d2$centered_weight, d2$height))
```

```
##               [,1]          [,2]
## [1,]  1.702946e-01 -3.638433e-18
## [2,] -3.638433e-18  4.096511e-03
```

Each entry in the variance-covariance matrix has been altered. In the regression output, we can see that the intercept has changed while the slope has remained the same.

(e)

```
#Frequentist regression confidence interval for the slope
confint(lm.1)
```

```
##                   2.5 %      97.5 %
## (Intercept) 110.1207774 117.6380098
## weight        0.8223315   0.9877267
```

```
confint(lm.2)
```

```
##                       2.5 %      97.5 %
## (Intercept)     154.0638974 155.1302878
## centered_weight   0.8223315   0.9877267
```

```r
#Bayesian regression posterior probability interval for the slope
post <- extract.samples(m4.3, n=1e4)

precis(post, prob = 0.95, digits = 5)
```

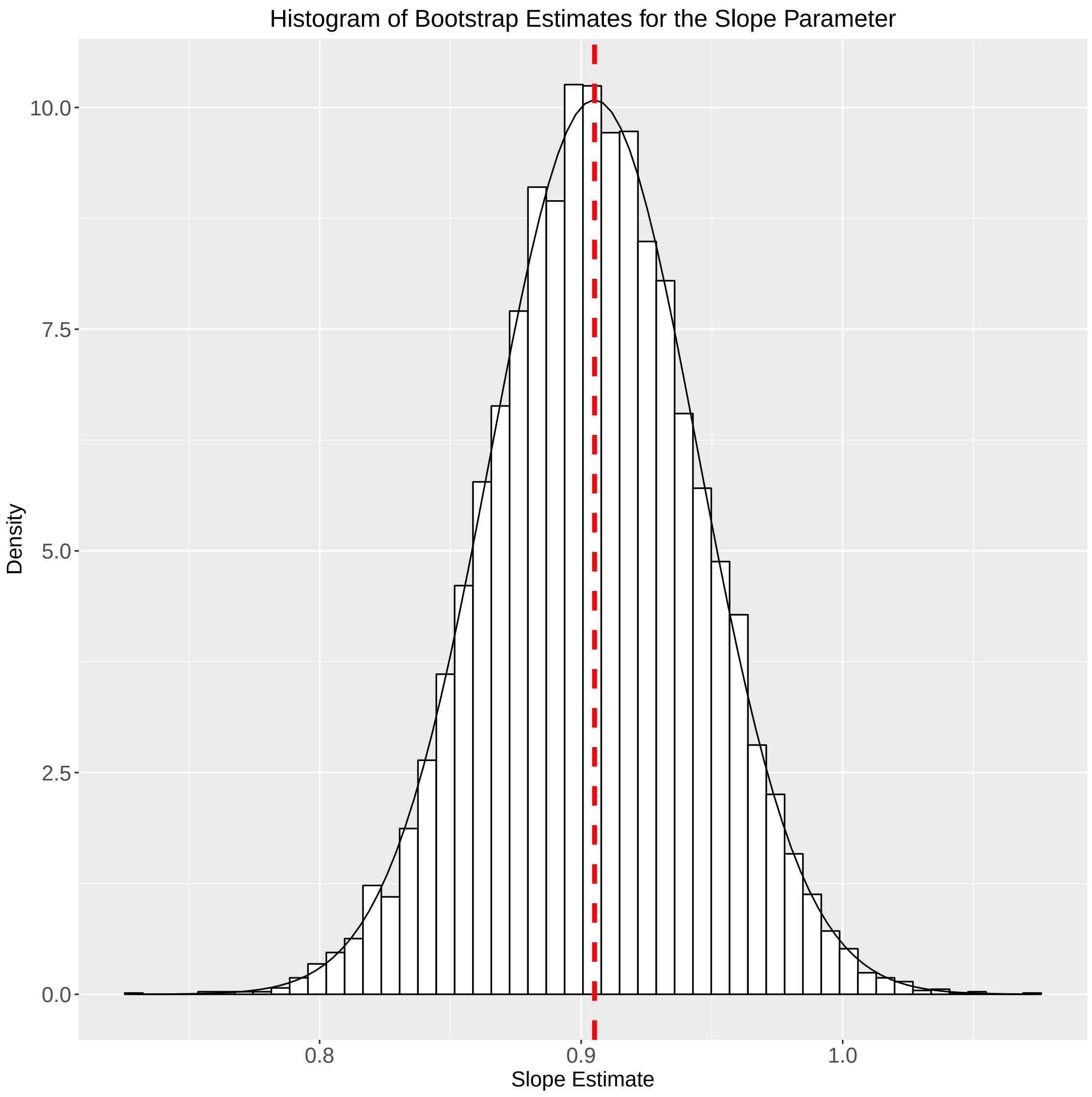
```
##             Mean  StdDev     |0.95     0.95|
## a     113.88343 1.91555 110.21521 117.64612
## b       0.90489 0.04213   0.82236   0.98618
## sigma   5.07235 0.19136   4.71182   5.45869
```

The 95% confidence interval for the slope can be thought of in this manner: if we were to take 100 different samples from the population, we would expect 95 of the regressions to yield an estimate for $\hat{\beta}_1$ that falls within the 95% confidence interval.

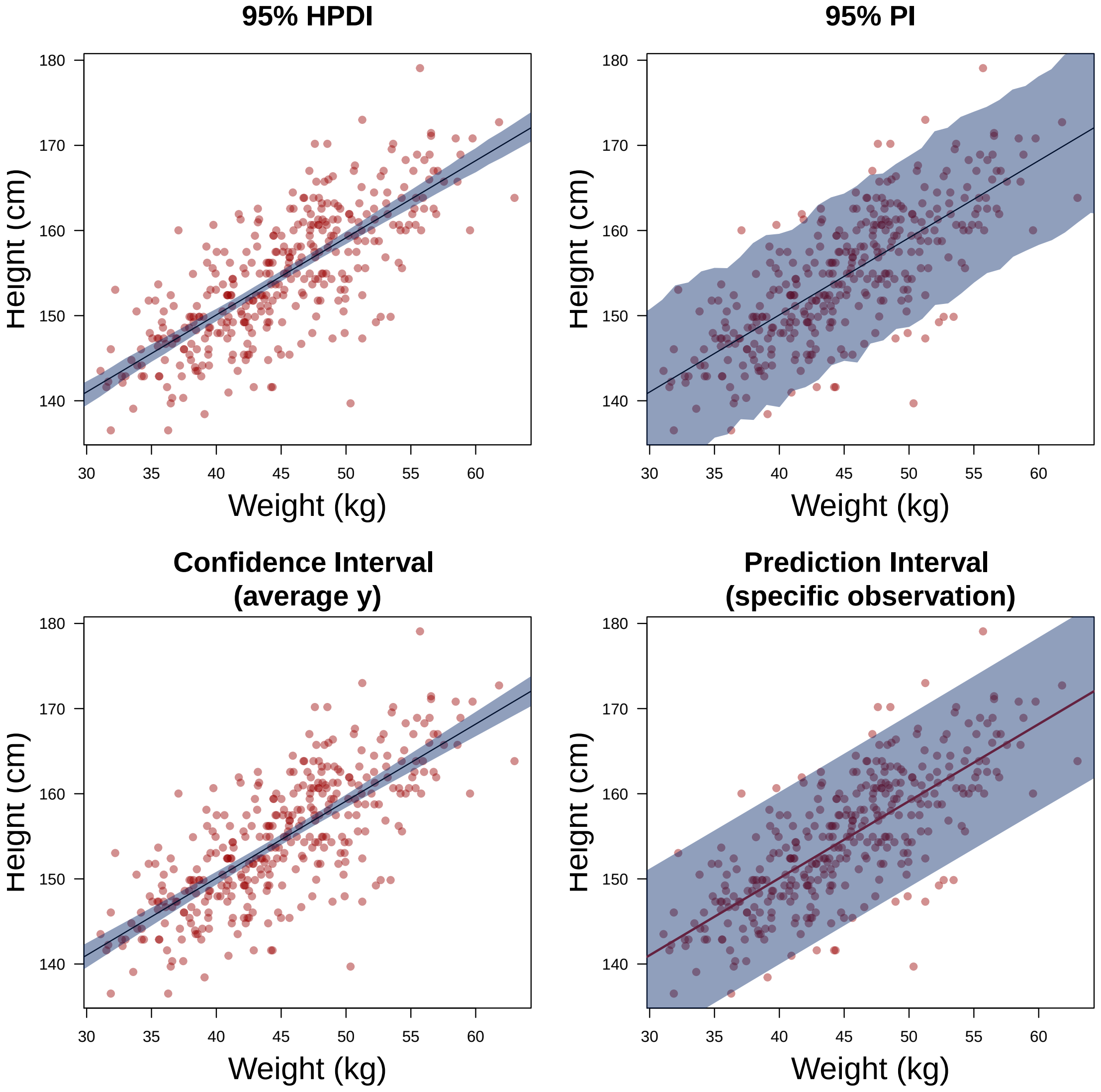
The 95% posterior probability interval for the slope represents the range within which we expect 95 out of 100 slopes sampled from the posterior distribution to fall.

The difference between the two lies within the interpretation of the slope parameter. Frequentist regression views $\hat{\beta}_1$ as a fixed parameter estimated from random data. However, Bayesian regression views $\hat{\beta}_1$ as a random variable with a sampling distribution (the posterior probability distribution).

```r
#Bootstrap interval
beta_hat <- rep(0, times = 10000)
for(i in 1:10000){
  data <- sample_n(d2, size = length(d2$height), replace = TRUE)
  lm <- lm(height ~ weight, data = data)
  beta_hat[i] <- lm$coefficients[2]
}

data <- data.frame("beta_hat" = beta_hat)

ggplot(data, aes(x=beta_hat)) +
  geom_histogram(aes(y = ..density..), bins = 50,
                 colour = "black",
                  fill = "white") +
  ggtitle("Histogram of Bootstrap Estimates for the Slope Parameter") +
  labs(x = "Slope Estimate", y = "Density") +
  stat_function(fun = dnorm, n = 101, args = list(mean = mean(beta_hat),
                                                  sd = sd(beta_hat))) +
  geom_vline(xintercept = mean(beta_hat), linetype = "dashed",
             color = "red", size = 1.5) +
  theme.info
```


Histogram of Bootstrap Estimates for the Slope Parameter
Density
10.0
7.5
5.0
2.5
0.0
0.8
0.9
1.0
Slope Estimate

(f) Let's plot the confidence & prediction intervals for both the frequentist and Bayesian models:


95% HPDI

95% PI

Confidence Interval (average y)

Prediction Interval (specific observation)

The plot in the top left demonstrates the densest 95% interval for the regression line.

The plot in the top right demonstrates the 95% confidence interval for a new y given each x value.

The bottom left plot demonstrates the 95% confidence interval for the average y value as x varies.

The bottom right plot demonstrates the 95% prediction interval for a new y given each x value.

The HPDI and Confidence Interval plots are relaying roughly the same information.

They each specify that the average y value for a given x should fall within that region with 95% probability.

The PI and Prediction Interval plots both show the region where the model expects to find 95% of heights in the population at each weight.