ARE 212 Midterm

Matthew Tadruno

March 14, 2018

Question 1

(Answers start below)

```
# Setup ----

# Packages
library(pacman)
library(knitr)
# p_load examples
p_load(dplyr, haven, readr, xtable, psych, magrittr)
p_load(ggplot2, extrafont, Matrix)

# Loading data
directory<-"/Users/matthewtarduno/Desktop/212/midterm/"
raw <-read_csv(paste0(directory, "data.csv"), col_types=cols())
```

```
# Cleaning Data ----

# note, I added a period dummy when I put the data into csv form.

# Using the dummy.code() function of psych
dummies<-as.data.frame(dummy.code(raw$state))
# Clean dummy names
names(dummies)<-gsub(' ', '\\.', names(dummies))


# Combine the two dataframes
data<-as.data.frame(c(raw, dummies))

#remove % sign
data$emmigrant.ratio<-substr(data$emmigrant.ratio, 1,
                             nchar(as.character(data$emmigrant.ratio))-1)
data$emmigrant.ratio<-as.numeric(data$emmigrant.ratio)
data$emmigrant.ratio<-data$emmigrant.ratio/100
```

```
# Functions ----

to_matrix <- function(the_df, vars) {
  mat <- the_df %>%
    select_(.dots = vars) %>%
    as.matrix()
  return(mat)
}
```

```r
# Function for OLS coefficient estimates
b_ols <- function(data, y_var, X_vars, intercept = TRUE) {
  require(dplyr)

  y <- to_matrix(the_df = data, vars = y_var)
  X <- to_matrix(the_df = data, vars = X_vars)

  # Use intercept arument
  if (intercept == T) {
    X <- cbind(1, X)
    colnames(X) <- c("intercept", X_vars)
  }

  # Calculate, return beta hat
  beta_hat <- solve(t(X) %*% X) %*% t(X) %*% y
  return(beta_hat)
}


# This function returns additional info about regression in df
ols <- function(data, y_var, X_vars, intercept = T) {

  # Turn data into matrices
  y <- to_matrix(data, y_var)
  X <- to_matrix(data, X_vars)

  # Add intercept based on arg
  if (intercept == T) X <- cbind(1, X)


  n <- nrow(X)
  k <- ncol(X)

  # Estimate coefficients usinf above function
  b <- b_ols(data, y_var, X_vars, intercept)

  # Calculate OLS residuals
  e <- y - X %*% b

  # Calculate sample var
  s2 <- (t(e) %*% e) / (n-k)

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

  se <- sqrt(s2 * diag(XX_inv))

  # Vector of _t_ statistics
  t_stats <- (b - 0) / se
```

```r
  # Calculate the p-values
  p_values = pt(q = abs(t_stats), df = n-k, lower.tail = F) * 2
  # table of results
  results <- data.frame(
    effect = rownames(b),
    coef = as.vector(b) %>% round(3),
    std_error = as.vector(se) %>% round(3),
    t_stat = as.vector(t_stats) %>% round(3),
    p_value = as.vector(p_values) %>% round(4)
  )
  return(results)
}
```

```r
# R2 Functions ----

demean <- function(N) {
  i <- matrix(data = 1, nrow = N)
  A <- diag(N) - (1/N) * i %*% t(i)
  return(A)
}


residuals <- function(data, y_var, X_vars, intercept = TRUE) {
  require(dplyr)
  y <- to_matrix(the_df = data, vars = y_var)
  X <- to_matrix(the_df = data, vars = X_vars)
  if (intercept == T) {
    X <- cbind(1, X)
    colnames(X) <- c("intercept", X_vars)
  }
  n <- nrow(X)
  resids <- (diag(n) - X %*% solve(t(X) %*% X) %*% t(X)) %*% y
  return(resids)
}


r2_ols <- function(data, y_var, X_vars) {

  y <- to_matrix(data, vars = y_var)
  X <- to_matrix(data, vars = X_vars)
  # Add intercept column to X
  X <- cbind(1, X)

  N <- nrow(X)
  K <- ncol(X)

  # Calculate the residuals
```

```r
  e <- residuals(data, y_var, X_vars)
  # Calculate the y_star (demeaned y)
  y_star <- demean(N) %*% y

  # Calculate r-squared values
  r2_uc  <- 1 - t(e) %*% e / (t(y) %*% y)
  r2     <- 1 - t(e) %*% e / (t(y_star) %*% y_star)
  r2_adj <- 1 - (N-1) / (N-K) * (1 - r2)

  return(c("r2_uc" = r2_uc, "r2" = r2, "r2_adj" = r2_adj))
}
```

Question 1 Part 1

Estimate model (1) via OLS.

These results do not match those from column 1 of the paper. The coefficient on log corn yield is **0.008** and the adjusted R-Squared value is **0.3212**. The coefficient on log corn plus wheat yield is **0.008** and the adjusted R-Squared value is **0.3215**.

```
# Question 1 ----

ols(data = data, y = "emmigrant.ratio", X = c("log.corn", "period"))

## Warning in s2 * diag(XX_inv): Recycling array of length 1 in array-vector
arithmetic is deprecated.
##  Use c() or as.vector() instead.

##       effect   coef std_error t_stat p_value
## 1 intercept  0.063     0.005 11.864  0.0000
## 2  log.corn  0.008     0.005  1.571  0.1214
## 3    period -0.037     0.007 -5.545  0.0000

r2_ols(data = data, y = "emmigrant.ratio", X = c("log.corn", "period"))

##     r2_uc        r2    r2_adj
## 0.8067434 0.3427066 0.3211560

#lm(data$emmigrant.ratio ~ data$log.corn + data$period)

ols(data = data, y = "emmigrant.ratio", X = c("log.corn.wheat", "period"))

## Warning in s2 * diag(XX_inv): Recycling array of length 1 in array-vector
arithmetic is deprecated.
##  Use c() or as.vector() instead.

##           effect   coef std_error t_stat p_value
## 1      intercept  0.063     0.005 11.523  0.0000
## 2 log.corn.wheat  0.008     0.005  1.580  0.1192
## 3         period -0.037     0.007 -5.529  0.0000

r2_ols(data = data, y = "emmigrant.ratio", X = c("log.corn.wheat", "period"))

##     r2_uc        r2    r2_adj
## 0.8068353 0.3430188 0.3214785

#lm(data$emmigrant.ratio ~ data$log.corn.wheat + data$period)

# Does not replicate column 1
```

Question 1 Part 2

Estimate model (1) via Fixed Effects using the Frisch Waugh transformation.

Using Frisch-Waugh, the coefficient on log corn yield is **-0.0123**, and the adjusted R-squared is **-0.00942**. The coefficient on log corn yield is **-0.00589**, and the adjusted R-squared is **-0.01469**. These results do not match those in column 3 of the paper.

```
# Question 1 Part 2 ----

# First reg log.corn on dummies, save the results (X2_transformed)
# then reg emmigrant.ratio on dummies, save the results (y_transformed)
# then run a new regression of y_transformed on X2_transformed

X1_vars<-c("period", colnames(dummies)[2:32])
#   index only 31, because we want to leave out one dummy
#   Aguascalientes will be the baseline

X2_transformed <- residuals(data, "log.corn", X1_vars)
y_transformed <- residuals(data, "emmigrant.ratio", X1_vars)

# Combine the two sets of residuals into a data.frame
Q2_data <- data.frame(
  y_transformed = y_transformed[,1], X2_transformed = X2_transformed[,1])

b_ols(data = Q2_data, y_var = "y_transformed",
      X_vars = "X2_transformed", intercept = F)

##                y_transformed
## X2_transformed   -0.01232232

r2_ols(data = Q2_data, y = "y_transformed", X = "X2_transformed")

##        r2_uc           r2       r2_adj
##  0.006604474  0.006604474 -0.009418035

#now with wheat...

X2_transformed <- residuals(data, "log.corn.wheat", X1_vars)
y_transformed <- residuals(data, "emmigrant.ratio", X1_vars)

# Combine the two sets of residuals into a data.frame
Q2_data <- data.frame(
  y_transformed = y_transformed[,1], X2_transformed = X2_transformed[,1])
```

```r
b_ols(data = Q2_data, y_var = "y_transformed",
      X_vars = "X2_transformed", intercept = F)

##                y_transformed
## X2_transformed  -0.005899236

r2_ols(data = Q2_data, y = "y_transformed", X = "X2_transformed")

##        r2_uc           r2       r2_adj
##  0.001413118  0.001413118 -0.014693122

# Again, this does not match the paper results
```

Question 1 Part 3

Estimate model (1) via OLS leaving out period fixed effects.

The coefficient on log corn yield is **0.005** and the adjusted R-Squared value is **-0.00452**. The coefficient on log corn plus wheat yield is **0.006** and the adjusted R-Squared value is **-0.00216**.

While the coefficients now match the paper, the R-squared values do not. It appears that the paper reported unadjusted rather than adjusted R-squared.

```
# Question 1 Part 3 ----

# Now leaving out the period fixed effect...

ols(data = data, y = "emmigrant.ratio", X = "log.corn")

## Warning in s2 * diag(XX_inv): Recycling array of length 1 in array-vector
arithmetic is deprecated.
##  Use c() or as.vector() instead.

##      effect  coef std_error t_stat p_value
## 1 intercept 0.046     0.005  8.708  0.0000
## 2  log.corn 0.005     0.006  0.846  0.4006

r2_ols(data = data, y = "emmigrant.ratio", X = "log.corn")

##        r2_uc           r2       r2_adj
##  0.709339543  0.011421808 -0.004523002

lm(data$emmigrant.ratio ~ data$log.corn)

##
## Call:
## lm(formula = data$emmigrant.ratio ~ data$log.corn)
##
## Coefficients:
##   (Intercept)  data$log.corn
##      0.046323       0.005406

ols(data = data, y = "emmigrant.ratio", X = "log.corn.wheat")

## Warning in s2 * diag(XX_inv): Recycling array of length 1 in array-vector
arithmetic is deprecated.
##  Use c() or as.vector() instead.
```

```
##           effect  coef std_error t_stat p_value
## 1      intercept 0.046     0.005  8.371  0.0000
## 2 log.corn.wheat 0.006     0.006  0.930  0.3561

r2_ols(data = data, y = "emmigrant.ratio", X = "log.corn.wheat")

##        r2_uc           r2       r2_adj
##  0.710023628  0.013748479 -0.002158803

lm(data$emmigrant.ratio ~ data$log.corn.wheat)

##
## Call:
## lm(formula = data$emmigrant.ratio ~ data$log.corn.wheat)
##
## Coefficients:
##         (Intercept)  data$log.corn.wheat
##            0.045815             0.005831

# These results are different that before, and now match the paper.
```

Question 1 Part 4

Estimate model (1) via F-W leaving out period fixed effects.

The coefficient on log corn yield is **-0.1172598** and the adjusted R-Squared value is **-0.00452**. The coefficient on log corn plus wheat yield is **-0.1131078** and the R-Squared value is **-0.00216**.

While the coefficients now match the paper, the R-squared values do not. It appears that the paper reported unadjusted rather than adjusted R-squared.

```
# Question 1 Part 4 ----


X1_vars<-colnames(dummies)[2:32]
#   index only 31, because we want to leave out one dummy
#   Aguascalientes will be the baseline
#   This time we leave out period

X2_transformed <- residuals(data, "log.corn", X1_vars)
y_transformed <- residuals(data, "emmigrant.ratio", X1_vars)

# Combine the two sets of residuals into a data.frame
Q4_data <- data.frame(
  y_transformed = y_transformed[,1], X2_transformed = X2_transformed[,1])

b_ols(data = Q4_data, y_var = "y_transformed",
      X_vars = "X2_transformed", intercept = F)

##                y_transformed
## X2_transformed    -0.1172598

r2_ols(data = Q4_data, y = "y_transformed", X = "X2_transformed")

##     r2_uc        r2    r2_adj
## 0.2994724 0.2994724 0.2881735

#now with wheat...

X2_transformed <- residuals(data, "log.corn.wheat", X1_vars)
y_transformed <- residuals(data, "emmigrant.ratio", X1_vars)

# Combine the two sets of residuals into a data.frame
Q4_data <- data.frame(
  y_transformed = y_transformed[,1], X2_transformed = X2_transformed[,1])

b_ols(data = Q4_data, y_var = "y_transformed",
      X_vars = "X2_transformed", intercept = F)
```

```
##                y_transformed
## X2_transformed    -0.1131078

r2_ols(data = Q4_data, y = "y_transformed", X = "X2_transformed")

##     r2_uc        r2    r2_adj
## 0.2375010 0.2375010 0.2252027

# These results are different than in part 2, and now match the paper.
```

Question 1 Part 5

What do you think happened here? And what are the consequences?

Not to be an R-squared maximizer, but the magnitude of change in the R-squared value suggests that the period fixed effects have significant explanitory power with regard to the emmigration rate. That is, there were likely time-varying factors that imfluenced emmigration across all states. Economic insuituion also suggests that they should be included in the model. It appears that the authors (knowingly or not) omitted these fixed effects when running their model. They also reported unadjusted rather than adjusted R-squared values, which made it difficult to see that the model had extrodinarily low predictive power when the period fixed effects were left out.

Question 2

(Answers start below)

```
# Part 2 (Normality of OLS) ----

p_load(dplyr, lfe, magrittr, parallel, lfe, ggplot2, ggthemes, viridis)

# theme ----

theme <- theme(
  legend.position = "bottom",
  panel.background = element_rect(fill = NA),
  panel.border = element_rect(fill = NA, color = "grey75"),
  axis.ticks = element_line(color = "grey85"),
  panel.grid.major = element_line(color = "grey95", size = 0.2),
  panel.grid.minor = element_line(color = "grey95", size = 0.2),
  legend.key = element_blank())

# Part A ----

#generate the population:

set.seed(22092008)
sample_size=100000
data_df <- data.frame(
  i = 1,
  x1 = rnorm(sample_size, 0, 5),
  x2 = rnorm(sample_size, 0, 5),
  e = rnorm(sample_size, 0, 5),
  eta = runif(sample_size, -8.66, 8.66))
# Calculate y = 7 + 0.5 x + e; drop 'e'
data_df %<>% mutate(y_a = 3 + 1*x1  - 2*x2 + e)
data_df %<>% mutate(y_b = 3 + 1*x1  - 2*x2 + eta)
```

```
# Functions used in simulation ----

# Function for OLS coefficient estimates
# Call to_matrix outside to make more efficient

b_ols <- function(y, X) {
  beta_hat <- solve(t(X) %*% X) %*% t(X) %*% y
  return(beta_hat)
}


# Function for a single iteration of the simulation
```

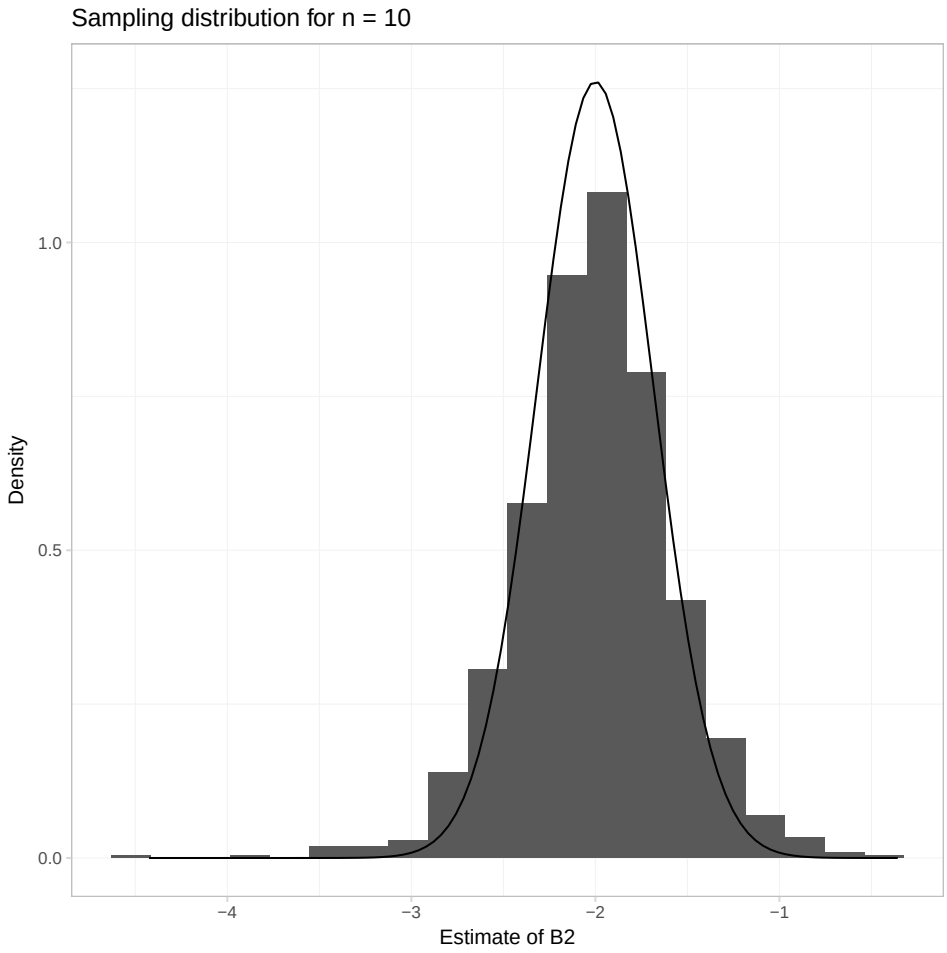
```r
one_iter <- function(iter, population, size) {
  sample_df <- sample_n(tbl = population, size)

  #OLs coefficients
  coef_ols <- b_ols(
    y = to_matrix(sample_df, "y_a"),
    X = to_matrix(sample_df, c("i", "x1", "x2")))

  # Create a data.frame to return
  coef_df <- data.frame(
    est    = as.vector(coef_ols),
    param  = c("int", "b1", "b2"),
    iter   = iter
  )
  print(coef_ols)
  return(coef_df)
}
```

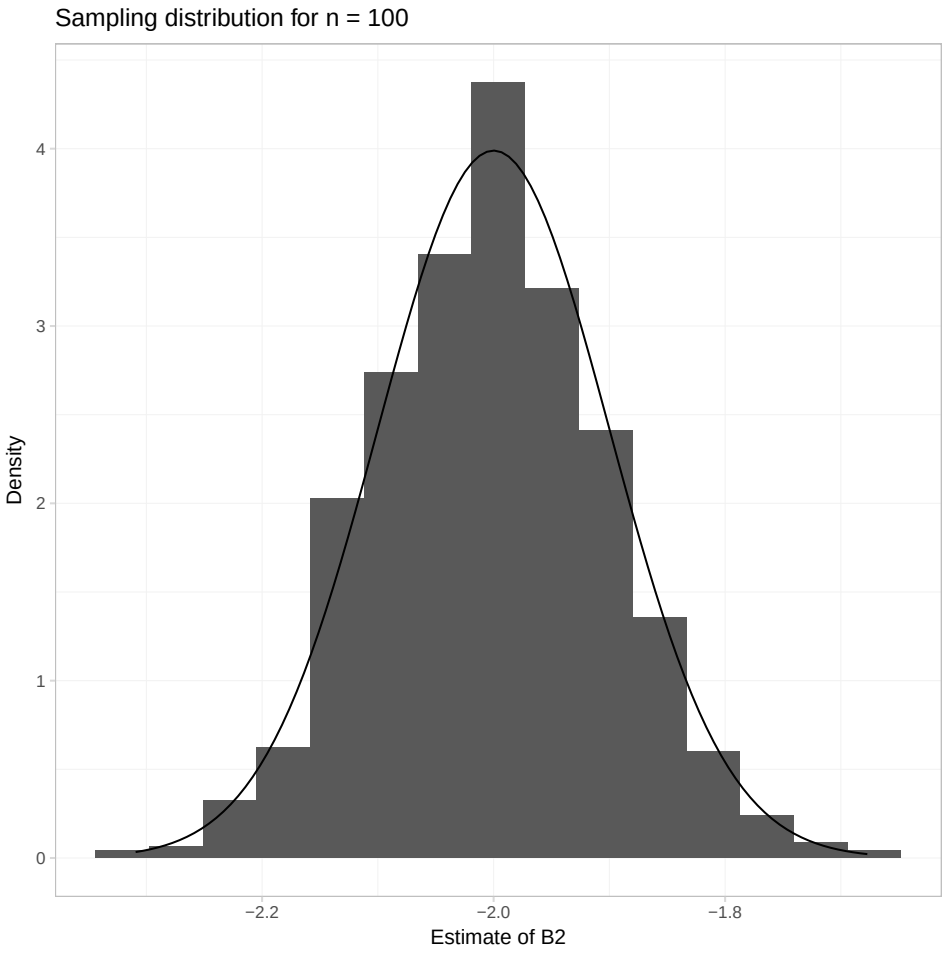
```r
# Clusters

cl <- makeCluster(4)
# Load packages on cluster
clusterEvalQ(cl, {
  library(dplyr)
  library(magrittr)
})
# Export our data and functions to the cluster
clusterExport(cl, "data_df")
clusterExport(cl, c("to_matrix", "b_ols", "one_iter"))
# Set seed in parallel
clusterSetRNGStream(cl, 12345)
```

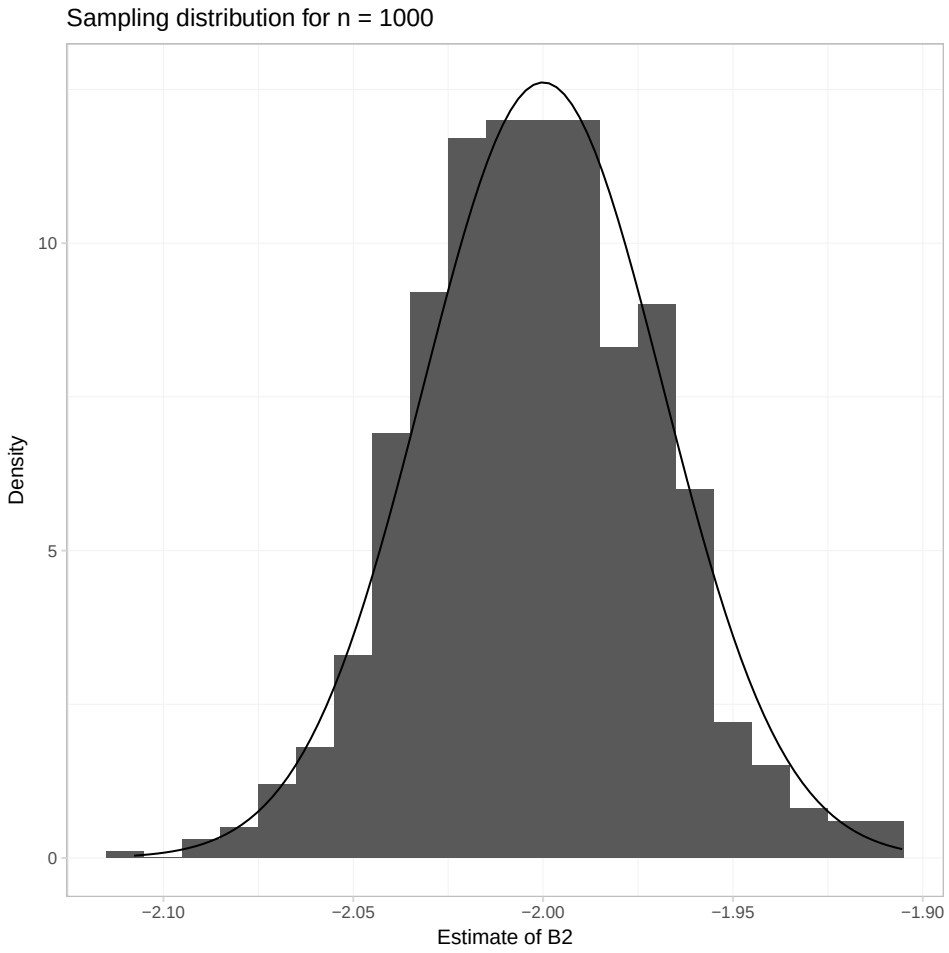
Question 2 Part A

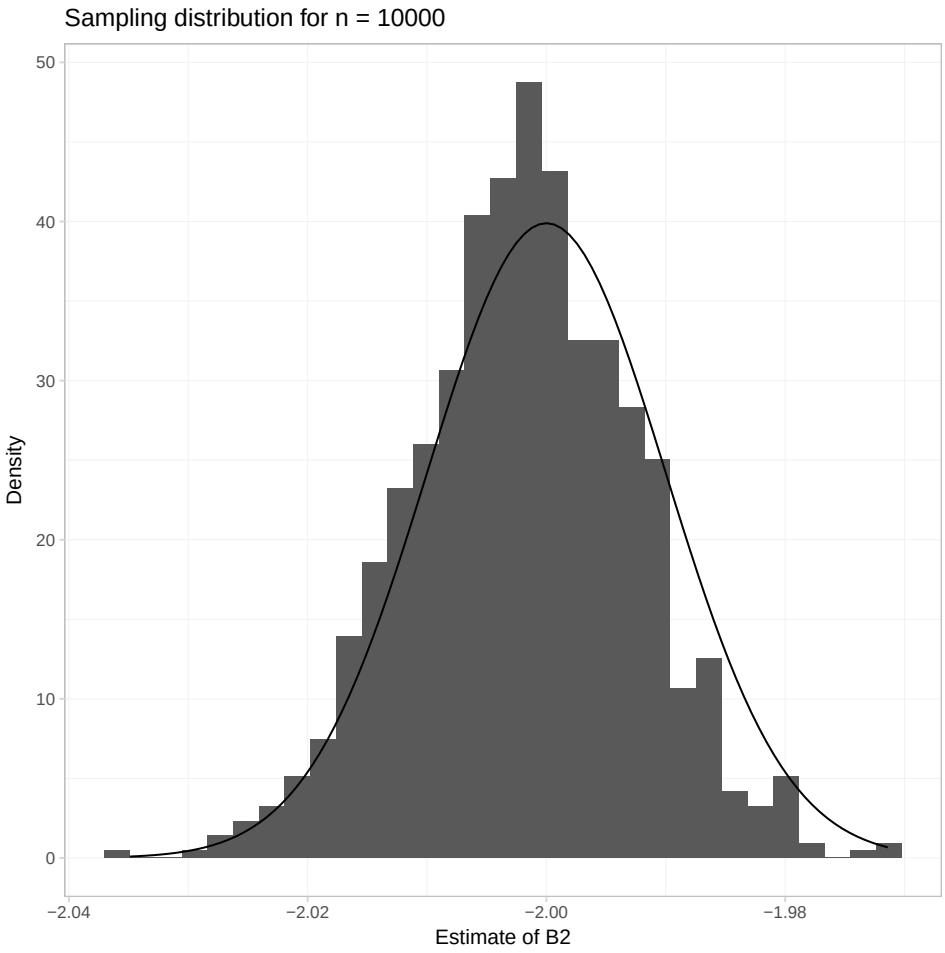
Here I loop over different sample sizes for n. For each n, I draw a sample of that size 1000 times from the population. For each draw I regress y_a on x_1, x_2, and a column of ones. I then plot the resulting distribution of estimates of β_2. The normal distribution overlayed had a mean of -2 and a variance of $\frac{(\sigma^2)}{v*N}$. Where σ is the variance of ϵ and v is the variance of x_2, as per section 4 of our notes. (Note: Discussion of results from parts A and B are on final page).

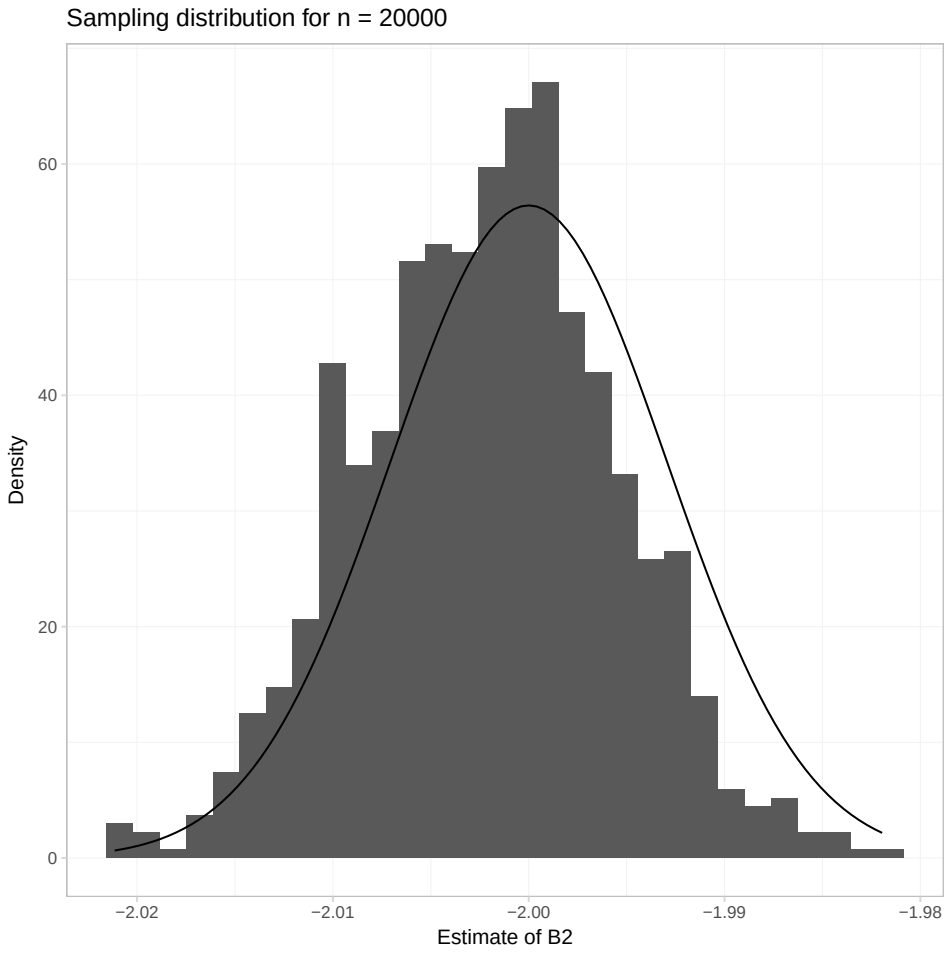
```
# repeating for all sample sizes:
n_list<-c(10, 100, 1000, 10000, 20000)
for (n in n_list) {
  sim_df <- parLapply(
    cl = cl,
    X = 1:1000,
    fun = one_iter,
    population = data_df, size=n) %>% bind_rows() %>% tbl_df()

  # Plotting ----

  plot_data<-subset(sim_df,  param == "b2")
  plot<-ggplot(plot_data, aes(est)) +
    geom_histogram(aes(y = ..density..), binwidth = 1/(n^(2/3))) +
    # Add normal dist with appropriate variance:
    stat_function(fun = dnorm, args = list(mean = -2, sd = 1/sqrt(n))) +
    xlab("Estimate of B2") +
    ylab("Density") +
    #add n to the title
    ggtitle(paste("Sampling distribution for n =", as.character(n)))+
    theme

  plot
  print(plot)
}
```


Sampling distribution for n = 10
Density
1.0
0.5
0.0
−4
−3
−2
−1
Estimate of B2


Sampling distribution for n = 100
Density
0
1
2
3
4
−2.2
−2.0
−1.8
Estimate of B2


Sampling distribution for n = 1000
Density
10
5
0
−2.10
−2.05
−2.00
−1.95
−1.90
Estimate of B2


Sampling distribution for n = 10000
Density
50
40
30
20
10
0
−2.04
−2.02
−2.00
−1.98
Estimate of B2

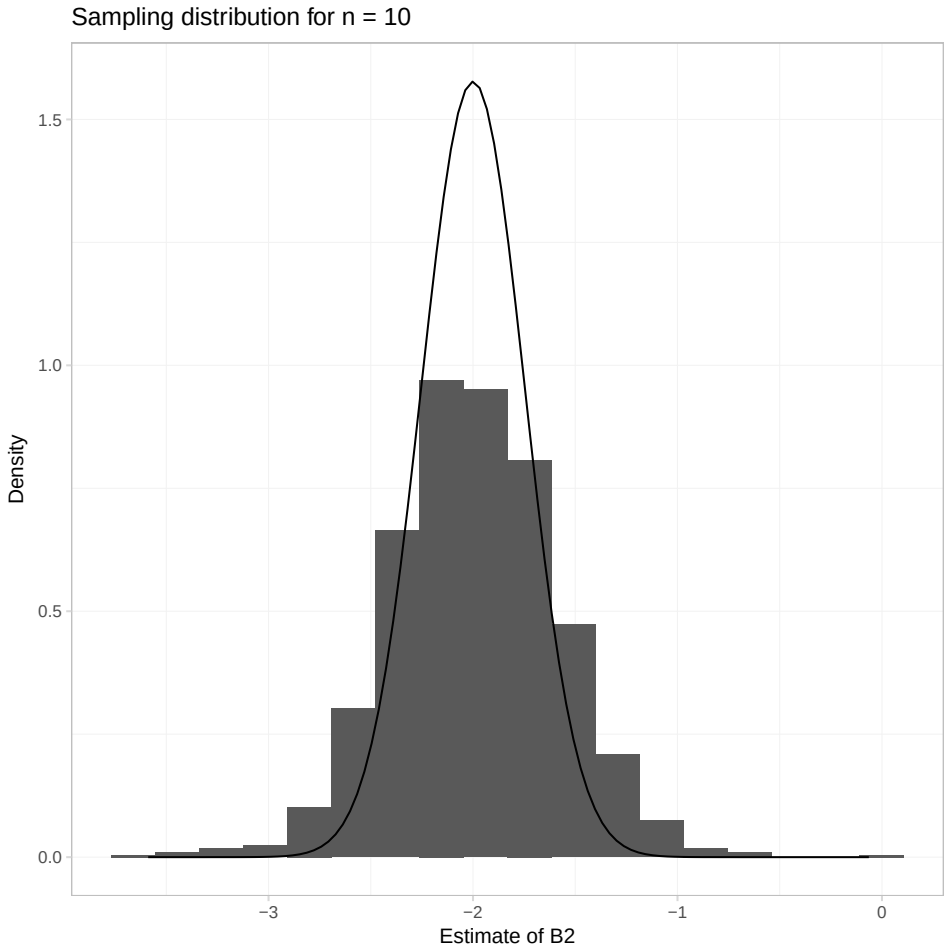

Sampling distribution for n = 20000
Density
60
40
20
0
−2.02
−2.01
−2.00
−1.99
−1.98
Estimate of B2

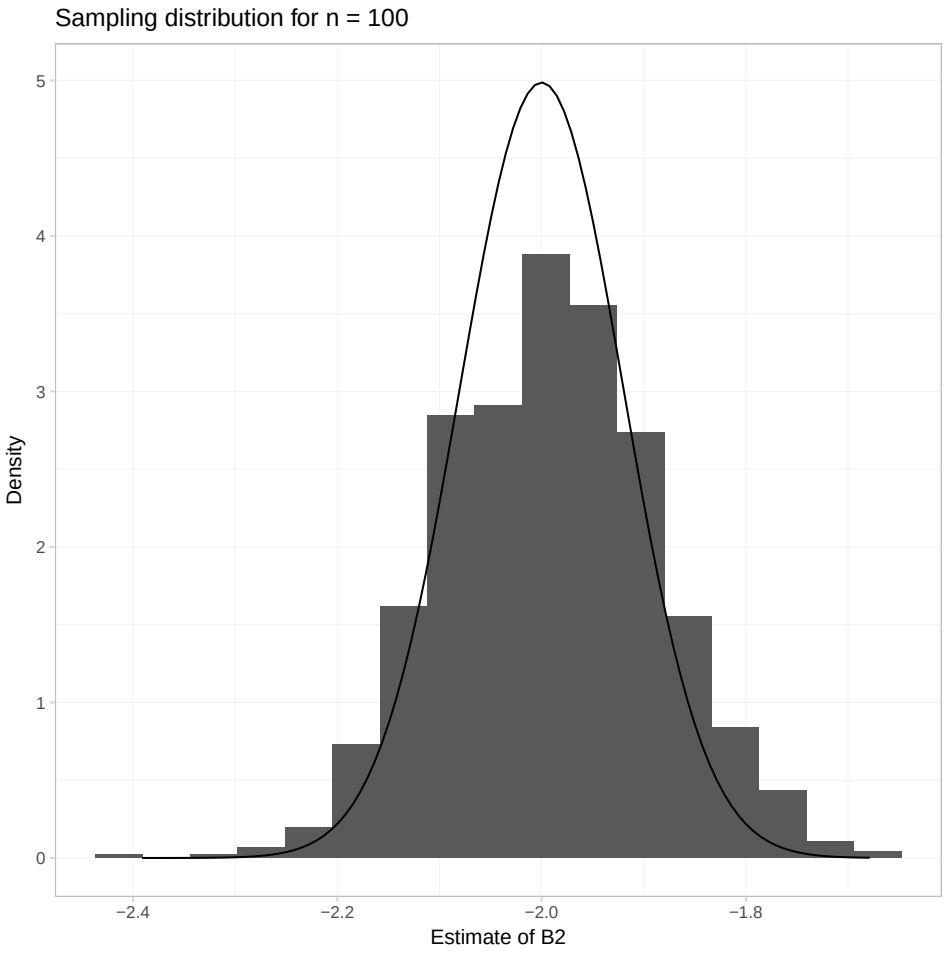
Question 2 Part B

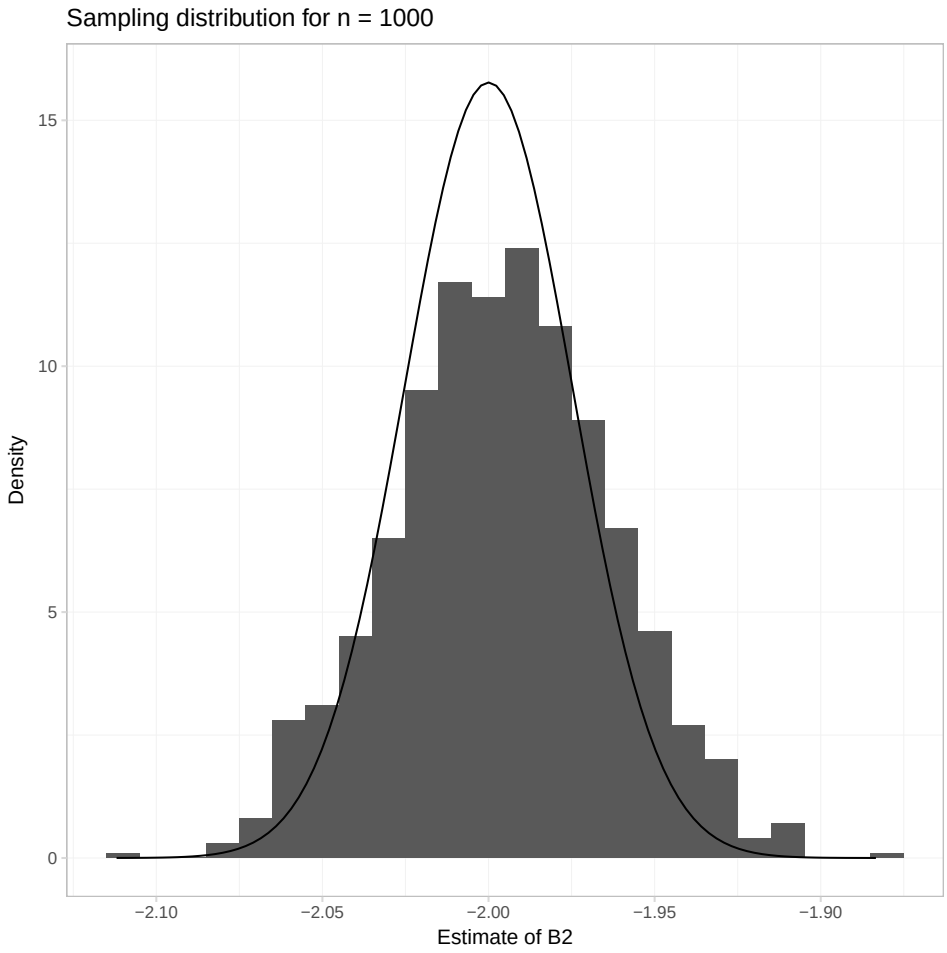
For this part we change the ols function to use the appropriate y, and adjust the variance of the normal distribution to reflect the fact that η has a different variance than ϵ.

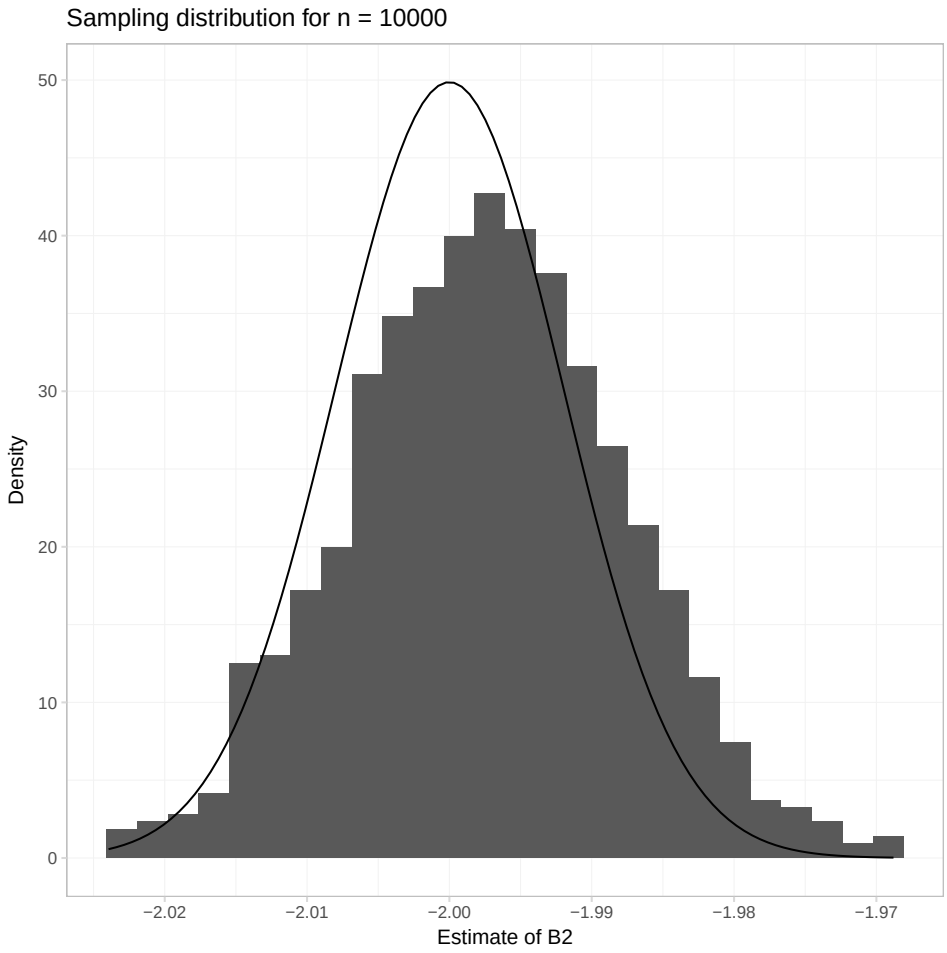
```
# Part B ----

# Just need to change up what we feed the ols function!
# Also note that the asymptotic variance is also different:
# v is going to be 1/12(8.66^2)=6.24963
# z1/v2 is 5/6.24963 = 0.8

one_iter <- function(iter, population, size) {
  sample_df <- sample_n(tbl = population, size)

  # Calculate the OLS coefficient
  coef_ols <- b_ols(
    y = to_matrix(sample_df, "y_b"),
    X = to_matrix(sample_df, c("i", "x1", "x2")))

  # Create a return a dataframe
  coef_df <- data.frame(
    est    = as.vector(coef_ols),
    param  = c("int", "b1", "b2"),
    iter   = iter
  )
  # Return the data.frame
  print(coef_ols)
  return(coef_df)
}


# repeating for all sample sizes:
n_list<-c(10, 100, 1000, 10000, 20000)
for (n in n_list) {
  sim_df <- parLapply(
    cl = cl,
    X = 1:1000,
    fun = one_iter,
    population = data_df, size=n) %>% bind_rows() %>% tbl_df()

  # Plot a separate histogram of the estimated B2's for each sample size:

  plot_data<-subset(sim_df,  param == "b2")
  plot<-ggplot(plot_data, aes(est)) +
    # Add normal dist with appropriate variance:
    geom_histogram(aes(y = ..density..), binwidth = 1/(n^(2/3))) +
```

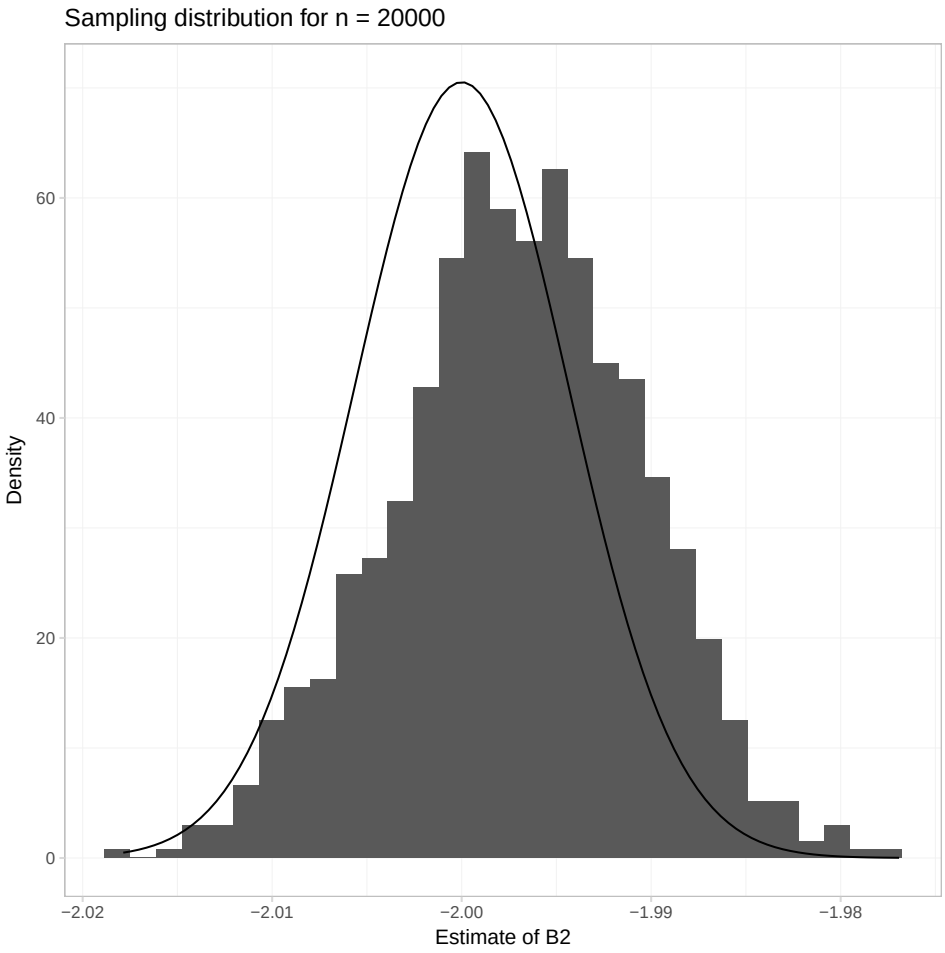
```
    stat_function(fun = dnorm, args = list(mean = -2, sd = .8/sqrt(n))) +
    xlab("Estimate of B2") +
    ylab("Density") +
    ggtitle(paste("Sampling distribution for n =", as.character(n)))+
    theme

  plot
  print(plot)
}
```




Sampling distribution for n = 100
Density
0
1
2
3
4
5
−2.4
−2.2
−2.0
−1.8
Estimate of B2


Sampling distribution for n = 1000
Density
15
10
5
0
−2.10
−2.05
−2.00
−1.95
−1.90
Estimate of B2


Sampling distribution for n = 10000
Density
50
40
30
20
10
0
−2.02
−2.01
−2.00
−1.99
−1.98
−1.97
Estimate of B2



We can see that as sample size increases, the distribution of β_2 approaches the overlayed normal distribution. This is true even in Part B, where the errors are not normall distributed. For a very large sample, however, the distribution doesn't line up exactly with the histogram. This is because the 'population' was itself a random sample, and thus the *true* value for the coefficient in the population from which we sample is not -2.