ECON312 Problem Set 1: question 3

Futing Chen, Will Parker

04/14/2020

Contents

Monte Carlo Simulations **2**
Part a) 2
Part b) 4

Nonparametric Bootstrap **6**
Part a) 6
Part b) 7

```r
library(tidyverse)
library(knitr)
```

Monte Carlo Simulations

Consider the model:

$$Y_i = X_i'\beta + U_i$$
$$U_i|X_i \overset{i.i.d}{\sim} N(0, \sigma^2)$$

Part a)

Define $\beta = (2,3)^T$, $\sigma^2 = 4$; generate $N = 10,000$ values for $X \in \mathbb{R}^2$. Using your value for σ^2 draw U's

```
set.seed(123456)
# doesn't appear that the distribution of X was specified
# I just used standard normal for x_1
N <- 10000


X_0 <- rep(1, N)
X_1 <-rnorm(n = N)

sigma <- 2
U <- rnorm(n = N, sd = sigma)

data <- tibble(X_0 = X_0,
               X_1 = X_1,
               U = U)

kable(head(data))
```

X_0	X_1	U
1	0.8337332	1.4791073
1	-0.2760478	3.5651206
1	-0.3550018	-3.0699699
1	0.0874874	0.0054147
1	2.2522557	0.6170447
1	0.8344601	4.3414702

Finally, compute the Y's

```
beta <- c(2,3)

data <- data %>%
  mutate(Y = X_0*beta[[1]] + X_1*beta[[2]] + U)

knitr::kable(head(data))
```

X_0	X_1	U	Y
1	0.8337332	1.4791073	5.980307
1	-0.2760478	3.5651206	4.736977
1	-0.3550018	-3.0699699	-2.134975
1	0.0874874	0.0054147	2.267877

X_0	X_1	U	Y
1	2.2522557	0.6170447	9.373812
1	0.8344601	4.3414702	8.844851

Estimate $\hat{\beta}$ and its standard errors from your data using standard OLS formulas.

We did the actual matrix calculations

$\hat{\beta} = (XX')^{-1}(XY)$

```
X <- as.matrix(tibble(int =1,
              X_1 = data$X_1))

Y <- data$Y

beta_n <- solve(t(X)%*%X)%*%t(X)%*%Y

kable(beta_n, col.names = "Beta")
```

	Beta
int	2.012333
X_1	2.962278

Standard errors

Under homoskedasticity which is given in the model,

$$V = XX'\hat{\sigma}^2$$

$$se(\hat{\beta}_k) = \sqrt{\frac{1}{n} diag(\hat{V})_k}$$

```
u <- Y - X%*%beta_n

u_sq <- as.vector(u *u)

sigma_sq_hat <- sum(u_sq)/N

V <- solve(t(X)%*%X)*sigma_sq_hat

se <- sqrt(diag(V))

kable(se, col.names = "standard error")
```

	standard error
int	0.0200031
X_1	0.0200324

Verifying with statistical software

```r
ols <- lm(Y ~ X_1, data)

summary(ols)
```

```
##
## Call:
## lm(formula = Y ~ X_1, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.5376 -1.3689 -0.0049  1.3556  7.5141
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.01233    0.02001   100.6   <2e-16 ***
## X_1          2.96228    0.02003   147.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2 on 9998 degrees of freedom
## Multiple R-squared:  0.6862, Adjusted R-squared:  0.6862
## F-statistic: 2.186e+04 on 1 and 9998 DF,  p-value: < 2.2e-16
```

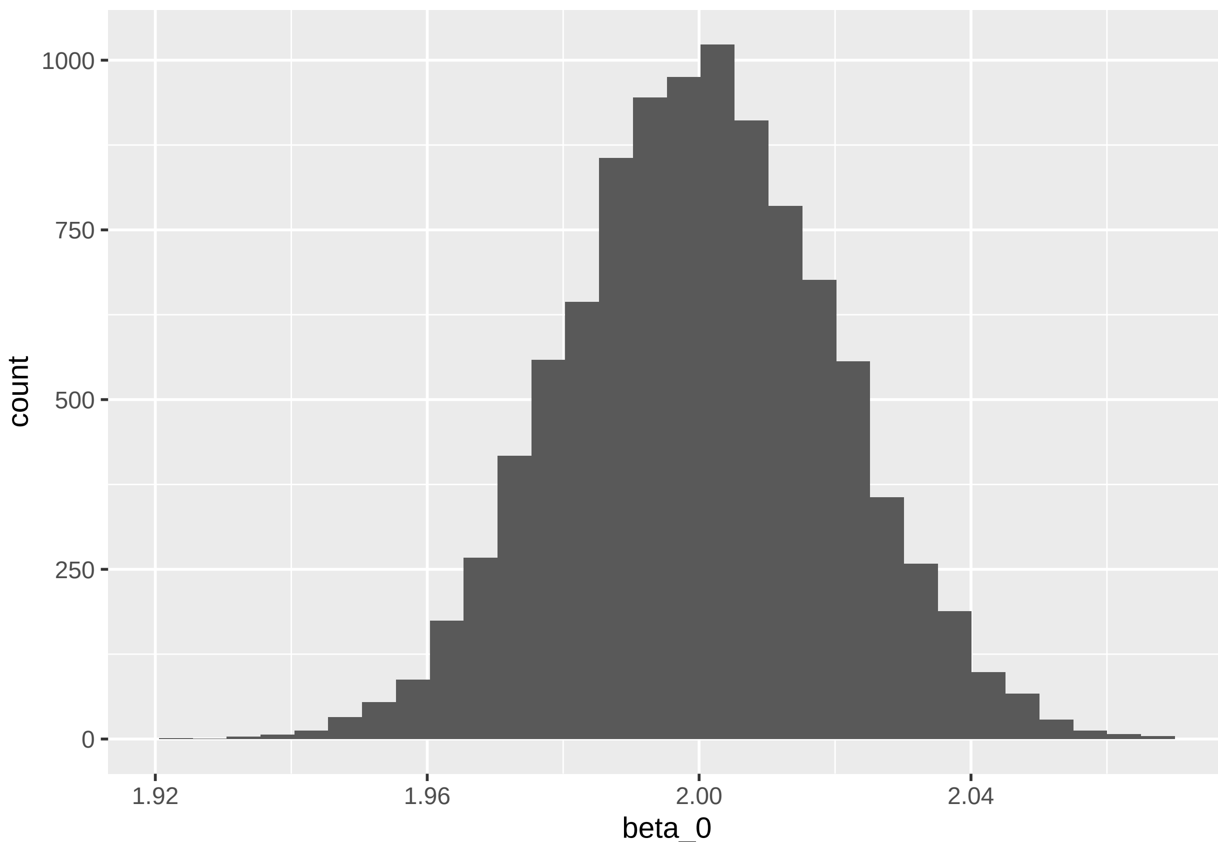
Part b)

Write a function to generate the $\hat{\beta}^{(s)}$s

```r
beta_sampler <- function(N = 10000, sigma = 2, beta =  c(2,3)){
  # we used standard normal for x_1
  X_0 <- rep(1, N)
  X_1 <-rnorm(n = N)


  U <- rnorm(n = N, sd = sigma)

  data <- tibble(X_0 = X_0,
                 X_1 = X_1,
                 U = U)
  data <- data %>%
    mutate(Y = X_0*beta[[1]] + X_1*beta[[2]] + U)

  X <- as.matrix(tibble(int =1,
              X_1 = data$X_1))

  Y <- data$Y

  beta_n <- solve(t(X)%*%X)%*%t(X)%*%Y

  return(beta_n)
}
```

```
S <- 10000

beta_0_list <- vector(mode = "numeric" , length = S)
beta_1_list <- vector(mode = "numeric" , length = S)

set.seed(123456)

for (k in seq(1:S)) {
  current_sample <- beta_sampler()

  beta_0_list[k] <- current_sample[[1]]
  beta_1_list[k] <- current_sample[[2]]

}
```

```
tibble(beta_0 = beta_0_list) %>%
  ggplot(aes(x = beta_0)) +
  geom_histogram()
```



Standard error of $\hat{(\beta_k)}$

First we need to justify that

$$\sqrt{\frac{1}{S}\sum_{s=1}^{S}(\hat{\beta}_k^{(s)})^2 - (\frac{1}{S}\sum_{s=1}^{S}\hat{\beta}_k^{(s)})^2} \xrightarrow{p} se(\hat{\beta}_k|X_1, X_2, ..X_n)$$

First note by WILLIN that because β_k is a random variable

$$\frac{1}{S}\sum_{s=1}^{S}(\hat{\beta}_k^{(s)}) \xrightarrow{p} E[\beta_k]$$

So then by the continuous mapping theorem

$$\frac{1}{S}\sum_{s=1}^{S}(\hat{\beta}_k^{(s)})^2 \xrightarrow{p} E[\beta_k^2]$$

And again by the continuous mapping theorem

$$\sqrt{\frac{1}{S}\sum_{s=1}^{S}(\hat{\beta}_k^{(s)})^2 - (\frac{1}{S}\sum_{s=1}^{S}\hat{\beta}_k^{(s)})^2} \xrightarrow{p} \sqrt{E[\beta_k^2] - E[\beta_k]^2} = se(\hat{\beta}_k|X_1, X_2, ..X_n)$$

computing $\sqrt{\hat{Var}[\hat{\beta}^{(s)}]}$

```
sd(beta_1_list)
```

```
## [1] 0.01970067
```

Very close to standard error produced by OLS procedure

Nonparametric Bootstrap

Part a)

```
set.seed(123456)
rct_sample <- function(N = 10000){
  U_1 <- rnorm(n = N)
  U_2 <- rnorm(n = N)

  assignment <- runif(n = N)

  sample <- tibble(U_1,
                   U_2,
                   assignment) %>%
    mutate(Y_1 = 5 + U_1,
           Y_0 = 2 + U_2,
           D = ifelse(assignment>0.5, 1, 0),
           beta = Y_1 - Y_0,
           Y = Y_0 + D*beta)


  return(sample)

}
```

```r
single_sample <- rct_sample()

summary(lm(Y ~ D, data = single_sample))
```

```
##
## Call:
## lm(formula = Y ~ D, data = single_sample)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2631 -0.6852  0.0085  0.6659  3.7108
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.00716    0.01407   142.7   <2e-16 ***
## D            3.01037    0.02002   150.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.001 on 9998 degrees of freedom
## Multiple R-squared:  0.6935, Adjusted R-squared:  0.6934
## F-statistic: 2.262e+04 on 1 and 9998 DF,  p-value: < 2.2e-16
```

OLS estimates are consistent

$$E[\beta_{OLS}] = \frac{Cov(Y,D)}{Var(D)} = \frac{Cov(DY_1 + (1-D)Y_0, D)}{Var(D)}$$

$$= \frac{E[DY_1 + D(1-D)Y_0] - E[DY_1 + (1-D)Y_0]E[D]}{E[D](1-E[D])}$$

$$= \frac{E[DY_1] - E[DY_1]E[D] - E[(1-D)Y_0]E[D]}{E[D](1-E[D])}$$

$$= \frac{E[Y_1|D=1]E[D](1-E[D]) - E[Y_0|D=0](1-E[D])E[D]}{E[D](1-E[D])}$$

$$= E[Y_1|D=1] - E[Y_0|D=0]$$

Since $D \perp\!\!\!\perp (Y_1, Y_0)$

$$E[\beta_{OLS}] = E[Y_1|D=1] - E[Y_0|D=0] = E[Y_1] - E[Y_0] = ATE = ATT = ATUT$$

Part b)

```r
N <- 10000
S <- 10000

beta_list <- vector(mode = "numeric", length = S)

for (k in seq(1:S)) {
    bootstrap <- single_sample %>%
      select(Y, D) %>%
      sample_n(size = N, replace = TRUE)

    model <- lm(Y ~ D, data = bootstrap)
```

```
    beta_list[[k]] <- model$coefficients[[2]]
}
```

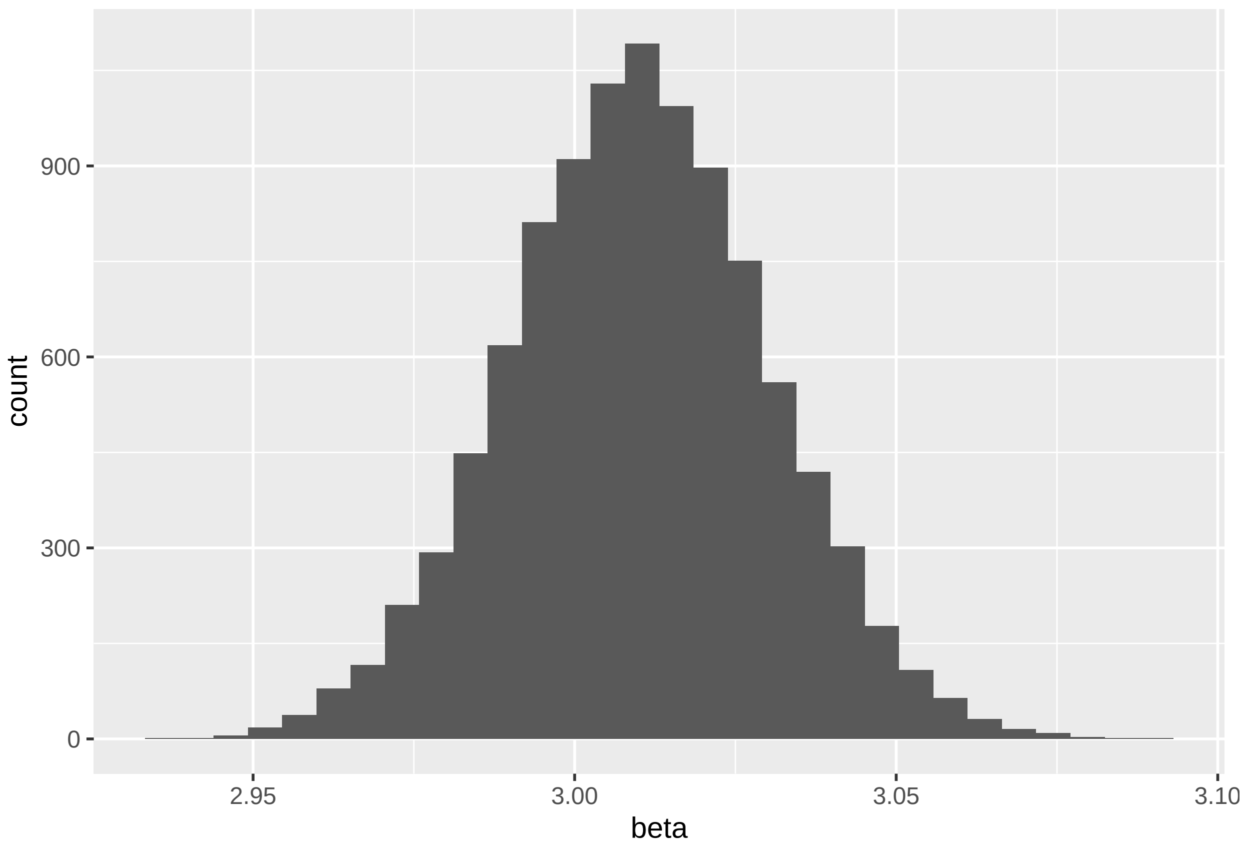
Standard error of $\hat{\beta}_k$

```
sd(beta_list)
```

```
## [1] 0.02015011
```

Histogram

```
tibble(beta = beta_list) %>%
  ggplot(aes(x = beta)) +
  geom_histogram()
```



bad boostrap

If Y and D were drawn independently from the sample, then treatment assignment would no longer be related to the observed outcome. We would be running a regression on (Y_1, D_0), (Y_1, D_1), (Y_0, D_0), and (Y_0, D_1) with equal probability (since $P(D = 1) = 0.5$ in our example). This would lead to a estimate of a treatment effect of 0.

```r
N <- 10000
S <- 10000

bad_boot_strap_list <- vector(mode = "numeric", length = S)

for (k in seq(1:S)) {
    Y <- sample(single_sample$Y, size = N, replace = TRUE)
    D <- sample(single_sample$Y, size = N, replace = TRUE)


    model <- lm(Y ~ D)


    bad_boot_strap_list[[k]] <- model$coefficients[[2]]
}

tibble(beta = bad_boot_strap_list) %>%
  ggplot(aes(x = beta)) +
  geom_histogram()
```