Homework 4 Solutions

7.17

(a)

To start with, let's write $\theta = r(\beta_1, \beta_2) = \beta_1 - \beta_2$. The key step is to derive an expression for $\sqrt{n}(\hat{\theta} - \theta)$. This is a linear function of the parameters, i.e., we can write $\theta = \mathbf{R}'\beta$ where $\mathbf{R} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$. Therefore, we have that

$$\begin{aligned} \sqrt{n}(\hat{\theta} - \theta) &= \mathbf{R}'\sqrt{n}(\hat{\beta} - \beta) \\ &\xrightarrow{d} \mathcal{N}(0, V) \end{aligned}$$

where

$$\begin{aligned} V &= \mathbf{R}'\mathbf{V}_\beta\mathbf{R} \\ &= \begin{bmatrix} 1 \\ -1 \end{bmatrix}' \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\ &= \begin{bmatrix} (V_{11} - V_{21}) & (V_{12} - V_{22}) \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} \\ &= V_{11} - V_{21} - V_{12} + V_{22} \end{aligned}$$

where V_{ij} denotes the element in the ith row and jth column in $\mathbf{V}_\beta$. This is the main theoretical result that we needed to show, but we would still need to estimate V in order to come up with a confidence interval. Before doing that, it is useful to note that we can write a 2×2 variance matrix, like $\mathbf{V}_\beta$ as

$$\mathbf{V}_\beta = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix} = \begin{bmatrix} V_{11} & \rho\sqrt{V_{11}}\sqrt{V_{22}} \\ \rho\sqrt{V_{11}}\sqrt{V_{22}} & V_{22} \end{bmatrix}$$

which holds because the diagonal elements of this matrix are variance, and the off diagonals are covariances (and recalling that $\text{cov}(X, Y) = \text{corr}(X, Y)\sqrt{\text{var}(X)}\sqrt{\text{var}(Y)}$ — which just holds from the definition of correlation). This suggests that,

$$\hat{V} = \hat{V}_{11} - 2\hat{\rho}\sqrt{\hat{V}_{11}}\sqrt{\hat{V}_{22}} + \hat{V}_{22}$$

which further implies that

$$\begin{aligned} \frac{\hat{V}}{n} &= \frac{\hat{V}_{11}}{n} - 2\hat{\rho}\frac{\sqrt{\hat{V}_{11}}}{\sqrt{n}}\frac{\sqrt{\hat{V}_{22}}}{\sqrt{n}} + \frac{\hat{V}_{22}}{n} \\ &= \text{se}(\hat{\beta}_1)^2 - 2\hat{\rho}\,\text{se}(\hat{\beta}_1)\,\text{se}(\hat{\beta}_2) + \text{se}(\hat{\beta}_2)^2 \end{aligned}$$

Finally, we can write down a 95% confidence interval as

$$\begin{aligned}\hat{C} &= \left[\hat{\theta} \pm 1.96\sqrt{\frac{\hat{V}}{n}}\right] \\ &= \left[\hat{\theta} \pm 1.96\sqrt{\text{se}(\hat{\beta}_1)^2 - 2\hat{\rho}\,\text{se}(\hat{\beta}_1)\,\text{se}(\hat{\beta}_2) + \text{se}(\hat{\beta}_2)^2}\right]\end{aligned}$$

where the first line is just the usual confidence interval (i.e., estimate plus or minus critical value times standard error), and the second equality plugs in the expression for $\hat{V}/n$ derived above.

(b)

No, it is not possible to calculate $\hat{\rho}$ from the information given in the problem. Besides the estimates of $\hat{\beta}_1$ and $\hat{\beta}_2$, the only other information that we have is about $\text{se}(\hat{\beta}_1)$ and $\text{se}(\hat{\beta}_2)$ — which does not tell us about their correlation.

(c)

I think the way to think about this problem is to think about the largest possible confidence interval given the information that we have. If this confidence interval does not include 0, then it would support the author's claim. As a side-comment, this is actually a really interesting question (at least in my view) because: on the one hand, you can immediately see that the 95% confidence interval for β_1 would not include the estimated value of β_2 (which is probably what the author is thinking), on the other hand, if you compute both confidence intervals for β_1 and β_2, they overlap (which would suggest that they are not different from each other). These are just heuristic arguments though, and our calculations above indicate that it is actually more complicated than either of these scenarios. Anyway…the widest possible confidence interval here will occur when $\hat{\rho} = -1$ (you can see this because it shows up in the negative term in the square root). Therefore, the widest possible confidence interval is given by

$$\begin{aligned}\hat{C}^{wide} &= \left[\hat{\theta} \pm 1.96\sqrt{\text{se}(\hat{\beta}_1)^2 + 2\text{se}(\hat{\beta}_1)\text{se}(\hat{\beta}_2) + \text{se}(\hat{\beta}_2)^2}\right] \\ &= \left[0.2 \pm 1.96\sqrt{4 \times 0.07^2}\right] \\ &= [-0.07, 0.47]\end{aligned}$$

This includes 0, which suggests that the author's claim is not correct. The information that we have from the problem does not necessarily imply that $\hat{\beta}_1$ and $\hat{\beta}_2$ are statistically different from each other.

7.28

(a)

We did part (a) on the previous homework, I am showing those results here so we can compare to them later

```
# read data
library(haven)
cps <- read_dta("cps09mar.dta")
```

```
# construct subset of white, male, Hispanic
data <- subset(cps, race==1 & female==0 & hisp==1)

# construct experience and wage
data$exp <- data$age - data$education - 6
data$wage <- data$earnings/(data$hours*data$week)

# run regression
Y <- log(data$wage)
X <- cbind(data$education, data$exp, data$exp^2/100, 1)
bet <- solve(t(X)%*%X)%*%t(X)%*%Y
round(bet,5)
```

```
         [,1]
[1,]  0.09045
[2,]  0.03538
[3,] -0.04651
[4,]  1.18521
```

```
# construct standard errors
ehat <- as.numeric(Y - X%*%bet)
Xe <- X*ehat
n <- nrow(data)
Omeg <- t(Xe)%*%Xe/n
XX <- t(X)%*%X/n
V <- solve(XX)%*%Omeg%*%solve(XX)
se <- sqrt(diag(V))/sqrt(n)
round(data.frame(beta=bet, se=se),5)
```

```
      beta      se
1  0.09045 0.00292
2  0.03538 0.00258
3 -0.04651 0.00530
4  1.18521 0.04608
```

(b)

$$\theta = \frac{\beta_1}{\beta_2 + 2\beta_3(10)/100} = \frac{\beta_1}{\beta_2 + \frac{1}{5}\beta_3}$$

```
thet <- bet[1]/(bet[2] + bet[3]/5)
thet
```

```
[1] 3.468335
```

(c)

We can use a Delta method argument for this. In particular, define

$$r(b) = \frac{b_1}{b_2 + \frac{1}{5}b_3}$$

and, therefore, we have that

$$\sqrt{n}(\hat{\theta} - \theta) = \nabla r(\beta)' \sqrt{n}(\hat{\beta} - \beta) + o_p(1)$$

where

$$\nabla r(\beta) = \left[\begin{array}{c} \frac{\partial r(b)}{\partial b_1} \\ \frac{\partial r(b)}{\partial b_2} \\ \frac{\partial r(b)}{\partial b_3} \\ \frac{\partial r(b)}{\partial b_4} \end{array}\right]\Bigg|_{b=\beta} = \left[\begin{array}{c} \frac{1}{b_2+\frac{1}{5}b_3} \\ -\frac{b_1}{(b_2+\frac{1}{5}b_3)^2} \\ -\frac{b_1}{5(b_2+\frac{1}{5}b_3)^2} \\ 0 \end{array}\right]\Bigg|_{b=\beta}$$

Thus, we have that

$$\sqrt{n}(\hat{\theta} - \theta) \xrightarrow{d} \mathcal{N}(0, \Gamma)$$

where

$$\Gamma = \nabla r(\beta)' \mathbf{V}_\beta \nabla r(\beta)$$

which we can estimate by

$$\hat{\Gamma} = \left[\begin{array}{c} \frac{1}{\hat{\beta}_2+\frac{1}{5}\hat{\beta}_3} \\ -\frac{\hat{\beta}_1}{(\hat{\beta}_2+\frac{1}{5}\hat{\beta}_3)^2} \\ -\frac{1}{5(\hat{\beta}_2+\frac{1}{5}\hat{\beta}_3)^2} \\ 0 \end{array}\right]' \hat{\mathbf{V}}_\beta \left[\begin{array}{c} \frac{1}{\hat{\beta}_2+\frac{1}{5}\hat{\beta}_3} \\ -\frac{\hat{\beta}_1}{(\hat{\beta}_2+\frac{1}{5}\hat{\beta}_3)^2} \\ -\frac{\hat{\beta}_1}{5(\hat{\beta}_2+\frac{1}{5}\hat{\beta}_3)^2} \\ 0 \end{array}\right]$$

and

$$\text{s.e.}(\hat{\theta}) = \frac{\sqrt{\hat{\Gamma}}}{\sqrt{n}}$$

```
# se(\hat{\theta})
r1 <- 1/(bet[2] + bet[3]/5)
r2 <- -bet[1]/(bet[2] + bet[3]/5)^2
r3 <- -bet[1]/(5*(bet[2] + bet[3]/5)^2)
r4 <- 0
r <- as.matrix(c(r1,r2,r3,r4))
Gamma <- t(r)%*%V%*%r
se_theta <- sqrt(Gamma)/sqrt(n)
se_theta
```

```
          [,1]
[1,] 0.2267341
```

(d)

```
# 90% confidence interval
ci_thet_L <- thet - 1.645*se_theta
ci_thet_U <- thet + 1.645*se_theta
paste0("[", round(ci_thet_L,3), ", ", round(ci_thet_U,3), "]")
```

```
[1] "[3.095, 3.841]"
```

(e)

```
# compute regression intervals and 95% confidence interval
x <- c(12,20,20^2/100,1)
m <- t(x)%*%bet
m
```

```
         [,1]
[1,] 2.792167
```

```
Vm <- t(x)%*%V%*%x
sem <- sqrt(Vm)/sqrt(n)
L <- m - 1.96*sem
U <- m + 1.96*sem
paste0("[",round(L,3),", ", round(U,3), "]")
```

```
[1] "[2.769, 2.815]"
```

Extra Question 1

```
set.seed(1234) # set seed for reproducibility

B <- 1000 # 1000 bootstrap iterations

boot_res <- list() # list to hold bootstrap results
for (b in 1:B) {
  # draw a sample
  index <- sample(1:n, replace=TRUE)
  boot_data <- data[index, ]
  Yb <- log(boot_data$wage)
  Xb <- cbind(boot_data$education, boot_data$exp, boot_data$exp^2/100, 1)
  betb <- solve(t(Xb)%*%Xb)%*%t(Xb)%*%Yb
  boot_res[[b]] <- betb
```

```
}

# bootstrap list to matrix
boot_res <- do.call(cbind, boot_res)
boot_res <- t(boot_res) # and take transpose for cov call below

# bootstrap estimate of asymptotic variance
Vb <- cov(boot_res)*n

# standard errors
seb <- sqrt(diag(Vb))/sqrt(n)

# compare to analytical standard errors computed above
round(data.frame(analytical=se, bootstrap=seb),5)
```

```
  analytical bootstrap
1    0.00292   0.00290
2    0.00258   0.00257
3    0.00530   0.00525
4    0.04608   0.04568
```

You can see that these are not exactly the same, but they are very close.

Extra Question 2

```
# function to run a single simulation
sim <- function() {
  # draw X1
  X1 <- rexp(n)

  # draw the error term
  e <- mixtools::rnormmix(n, lambda=c(.5,.5), mu=c(-2,2), sigma=c(1,1))

  ## TODO: construct Y
  Y <- b0 + b1*X1 + e

  ## TODO: use X1 and Y to estimate bet0 and bet1
  X <- cbind(1,X1)
  bet <- solve(t(X)%*%X)%*%t(X)%*%Y
  bet1 <- bet[2,1]

  # TODO: return estimated value of bet1
  bet1
}

# function to run many simulations
# @param n_sims is the number of simulations to run
```

```r
run_mc <- function(n_sims=1000) {

  # run n_sims simulations and store in a vector
  mc_res <- sapply(1:n_sims, function(s) {
    sim()
  })

  # print number of observations
  cat("n = ", n, "....\n")

  # print the mean of b1
  cat("mean b1  : ", mean(mc_res), "\n")

  # print the variance of b1
  cat("var b1   : ", var(mc_res), "\n")
}
```

```r
# run the simulations
# set values of parameters and number of observations
set.seed(1234) # so can reproduce
b0 <- 0
b1 <- 1

n <- 2
run_mc()
```

```
n =  2 ....
mean b1  :  2.506428
var b1   :  4841.861
```

```r
n <- 10
run_mc()
```

```
n =  10 ....
mean b1  :  1.036217
var b1   :  1.074335
```

```r
n <- 50
run_mc()
```

```
n =  50 ....
mean b1  :  0.9884143
var b1   :  0.122526
```

```
n <- 100
run_mc()
```

```
n =  100 ....
mean b1   :  1.004076
var b1    :  0.05406661
```

```
n <- 500
run_mc()
```

```
n =  500 ....
mean b1   :  1.006865
var b1    :  0.01053563
```

It looks like our theory is holding here. $\hat{\beta}_1$ appears to be unbiased — recall unbiasedness is a finite sample property — so this should hold for all values of n (the only case where there are issues is when $n = 2$; in this case, you can see that the variance is extremely high, and I think that we are not doing enough simulations to see that it is actually unbiased in this case). The other interesting thing to note is that, as expected, the variance of $\hat{\beta}_1$ is decreasing for larger sample sizes.

Extra Question 3

(a)

Given our discussion in class (and given that $\mathbb{H}_0$ is true here), we would expect/hope to reject about 5% of the time.

(b)

```
# function to run a single simulation
sim <- function() {
  # draw X1
  X1 <- rexp(n)

  # draw the error term
  e <- mixtools::rnormmix(n, lambda=c(.5,.5), mu=c(-2,2), sigma=c(1,1))

  ## construct Y
  Y <- b0 + b1*X1 + e

  ## estimate bet1 and V and construct t-stat
  X <- cbind(1,X1)
  bet <- solve(t(X)%*%X)%*%t(X)%*%Y
  ehat <- as.numeric(Y-X%*%bet)
  Xe <- X*ehat
  XX <- t(X)%*%X/n
  Omeg <- t(Xe)%*%Xe/n
```

```
  V <- solve(XX)%*%Omeg%*%solve(XX)
  bet1 <- bet[2,1]
  t_stat <- sqrt(n)*(bet1 - H0)/sqrt(V[2,2])

  # return whether or not reject
  1*(abs(t_stat) > qnorm(.975))
}

# function to run many simulations
# @param n_sims is the number of simulations to run
run_mc <- function(n_sims=1000) {

  # run n_sims simulations and store in a vector
  mc_res <- sapply(1:n_sims, function(s) {
    sim()
  })

  # print rejection probability
  cat("rej. prob  : ", mean(mc_res), "\n")
}


# run the simulations
# set values of parameters and number of observations
set.seed(1234)
b0 <- 0
b1 <- 1
H0 <- 1
n <- 100

run_mc()
```

```
rej. prob  :  0.071
```

We reject 7.1% of the time here. This looks like we are slightly over-rejecting (relative to the fraction of time that we'd like to), but this seems to at least be working pretty well.

(c)

```
# n=10
n <- 10
run_mc()
```

```
rej. prob  :  0.256
```

```
# n=50
n <- 50
run_mc()
```

```
rej. prob  :  0.1
```

```
# n=500
n <- 500
run_mc()
```

```
rej. prob  :  0.052
```

```
# n=1000
n <- 1000
run_mc()
```

```
rej. prob  :  0.045
```

These results are quite interesting. When $n = 10$, we reject $\mathbb{H}_0$ 25.6% of the time — in other words, despite being true, we reject the null about 25% of the time when we only have 10 observations. This suggests that our asymptotic approximation arguments for the limiting distribution of $\sqrt{n}(\widehat{\beta} - \beta)$ are not working very well when $n = 10$. This should not be surprising though because n is quite small here.

The performance of inference procedure is better, though we still over-reject, when $n = 50$. By the time $n = 500$ or $n = 1000$, it looks like our inference procedure is working quite well.

(d)

In this case $\mathbb{H}_0$ is false, so we'd like to reject $\mathbb{H}_0$. We expect to have more power (i.e., be able to reject a false null) as the number of observations increases.

```
set.seed(1234)
b0 <- 0
b1 <- 1
H0 <- 0

# n=10
n <- 10
run_mc()
```

```
rej. prob  :  0.439
```

```
# n=50
n <- 50
run_mc()
```

```
rej. prob  :  0.844
```

```r
# n=100
n <- 100
run_mc()
```

```
rej. prob  :  0.987
```

```r
# n=500
n <- 500
run_mc()
```

```
rej. prob  :  1
```

```r
# n=1000
n <- 1000
run_mc()
```

```
rej. prob  :  1
```

This is exactly what we find. When $n = 10$, we reject only 44% of the time; when $n = 50$, we reject 84% of the time; when $n = 100$, we reject almost 99% of the time; and for higher values of n, we reject 100% of the time.

If you are interested, it would interesting to experiment with different values of `b1` and/or `H0` here and also see how that affects the power of the test.