Homework 3 Solutions

7.7

(a)

β is defined as the coefficient of the linear projection of Y^* on X. Thus, $\beta = \mathrm{E}[XX']^{-1}\mathrm{E}[XY^*]$. Now, let's define

$$\tilde{\beta} = \underset{b}{\operatorname{argmin}}\ \mathrm{E}[(Y - X'b)^2]$$

so that $\tilde{\beta}$ is the coefficient from the linear projection of Y on X. Solving this, we get that

$$\begin{aligned}
\tilde{\beta} &= \mathrm{E}[XX']^{-1}\mathrm{E}[XY] \\
&= \mathrm{E}[XX']^{-1}\mathrm{E}[X(Y^* + u)] \\
&= \mathrm{E}[XX']^{-1}\mathrm{E}[XY^*] + \mathrm{E}[XX']^{-1}\underbrace{\mathrm{E}[Xu]}_{=0} \\
&= \beta
\end{aligned}$$

Thus, $\tilde{\beta} = \beta$.

I think the above is the correct answer to the question, but there is one more thing that is worth pointing out. As in the problem, let's define $\hat{\beta}$ as the estimate that comes from running a regression of Y on X, and additionally define $\hat{\beta}^*$ as the (infeasible) regression coefficient that you would get if you could run the regression of Y^* on X. Note that

$$\hat{\beta}^* = \left(\frac{1}{n}\sum_{i=1}^{n} X_i X_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} X_i Y_i^*$$

and

$$\begin{aligned}
\hat{\beta} &= \left(\frac{1}{n}\sum_{i=1}^{n} X_i X_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} X_i Y_i \\
&= \left(\frac{1}{n}\sum_{i=1}^{n} X_i X_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} X_i (Y_i^* + u_i) \qquad (1)
\end{aligned}$$

so, in general, $\hat{\beta} \neq \hat{\beta}^*$; that is, if we were to observe Y_i^*, we would not get numerically estimates from the regression of Y on X as from the regression of Y^* on X.

(b)

From Equation 1, we can write

$$\begin{aligned}
\hat{\beta} &= \left(\frac{1}{n}\sum_{i=1}^{n} X_i X_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} X_i Y_i^* + \left(\frac{1}{n}\sum_{i=1}^{n} X_i X_i'\right)^{-1} \frac{1}{n}\sum_{i=1}^{n} X_i u_i \\
&\xrightarrow{p} \mathrm{E}[XX']^{-1}\mathrm{E}[XY^*] + 0 \\
&= \beta
\end{aligned}$$

where the second equality holds by the law of large numbers and the continuous mapping theorem. This implies that, despite the measurement error, $\hat{\beta}$ is consistent for β.

(c)

Plugging in $Y_i^* = X_i'\beta + e_i$ into Equation 1 and multiplying by $\sqrt{n}$, we have that

$$\begin{aligned}\sqrt{n}(\hat{\beta} - \beta) &= \left(\frac{1}{n}\sum_{i=1}^n X_i X_i'\right)^{-1} \frac{1}{\sqrt{n}}\sum_{i=1}^n X_i(e_i + u_i) \\ &= \mathrm{E}[XX']^{-1}\frac{1}{\sqrt{n}}\sum_{i=1}^n X_i(e_i+u_i) + o_p(1) \\ &\xrightarrow{d} N(0, \mathrm{E}[XX']^{-1}\Omega\mathrm{E}[XX']^{-1})\end{aligned}$$

where

$$\Omega = \mathrm{E}[XX'(e+u)^2]$$

This is related, but different, from the case without measurement error; recall that, in that case $\Omega = \mathrm{E}[XX'e^2]$.

Altogether, this suggests that, when there is this relatively simple kind of measurement error in the outcome, using the measured-with-error outcome still delivers consistent estimates of β, but the asymptotic variance changes; it is likely to be bigger.

7.14

(a)

$$\hat{\theta} = \hat{\beta}_1\hat{\beta}_2$$

where $\hat{\beta}_1$ and $\hat{\beta}_2$ come from the regression of Y on X_1 and X_2.

(b)

First, notice that our usual arguments imply that

$$\sqrt{n}\begin{pmatrix}\hat{\beta}_1 - \beta_1 \\ \hat{\beta}_2 - \beta_2)\end{pmatrix} \xrightarrow{d} N(0, \mathbf{V}_\beta)$$

where $\mathbf{V}_\beta = \mathrm{E}[XX']^{-1}\mathbf{\Omega}\mathrm{E}[XX']^{-1}$, where we take $X = (X_1, X_2)'$ and where $\mathbf{\Omega} = \mathrm{E}[XX'e^2]$. Note that $\mathbf{V}_\beta$ is a 2×2 variance matrix.

Next, notice that we can write $\theta = r(\beta_1, \beta_2)$ and $\hat{\theta} = r(\hat{\beta}_1, \hat{\beta}_2)$ where $r(b_1, b_2) = b_1 b_2$. Moreover, using a mean value theorem argument, we have that

$$r(\hat{\beta}_1, \hat{\beta}_2) = r(\beta_1, \beta_2) + \nabla r(\bar{\beta}_1, \bar{\beta}_2)'\begin{pmatrix}\hat{\beta}_1 - \beta_1 \\ \hat{\beta}_2 - \beta_2\end{pmatrix}$$

where

$$\nabla r(\bar{\beta}_1, \bar{\beta}_2) := \begin{bmatrix}\frac{\partial r(b_1,b_2)}{\partial b_1} \\ \frac{\partial r(b_1,b_2)}{\partial b_2}\end{bmatrix}\Bigg|_{b_1=\bar{b}_1, b_2=\bar{b}_2} = \begin{bmatrix} b_2 \\ b_1\end{bmatrix}\Bigg|_{b_1=\bar{b}_1, b_2=\bar{b}_2} = \begin{bmatrix}\bar{b}_1 \\ \bar{b}_2\end{bmatrix}$$

This implies that

$$\begin{aligned}
\sqrt{n}(\hat\theta - \theta) &= \begin{bmatrix} \bar b_1 \\ \\ \bar b_2 \end{bmatrix}' \sqrt{n} \begin{pmatrix} \hat\beta_1 - \beta_1 \\ \hat\beta_2 - \beta_2 \end{pmatrix} \\
&= \begin{bmatrix} b_1 \\ \\ b_2 \end{bmatrix}' \sqrt{n} \begin{pmatrix} \hat\beta_1 - \beta_1 \\ \hat\beta_2 - \beta_2 \end{pmatrix} + \underbrace{\left(\begin{bmatrix} \bar b_1 \\ \\ \bar b_2 \end{bmatrix} - \begin{bmatrix} b_1 \\ \\ b_2 \end{bmatrix} \right)'}_{=o_p(1)} \underbrace{\sqrt{n} \begin{pmatrix} \hat\beta_1 - \beta_1 \\ \hat\beta_2 - \beta_2 \end{pmatrix}}_{=O_p(1)} \\
&= \begin{bmatrix} b_1 \\ \\ b_2 \end{bmatrix}' \sqrt{n} \begin{pmatrix} \hat\beta_1 - \beta_1 \\ \hat\beta_2 - \beta_2 \end{pmatrix} + o_p(1) \\
&\xrightarrow{d} N(0, V)
\end{aligned}$$

where

$$V = \begin{bmatrix} b_1 \\ \\ b_2 \end{bmatrix}' \mathbf{V}_\beta \begin{bmatrix} b_1 \\ \\ b_2 \end{bmatrix}$$

(c)

To calculate a 95% confidence interval, the main step is to estimate V. The natural estimate is given by

$$\begin{bmatrix} \hat b_1 \\ \\ \hat b_2 \end{bmatrix}' \hat{\mathbf{V}}_\beta \begin{bmatrix} \hat b_1 \\ \\ \hat b_2 \end{bmatrix}$$

and where we use the usual estimate of $\mathbf{V}_\beta$ that is given by

$$\hat{\mathbf{V}}_\beta = \left(\frac{1}{n} \sum_{i=1}^n X_i X_i' \right)^{-1} \frac{1}{n} \sum_{i=1}^n X_i X_i' \hat e_i^2 \left(\frac{1}{n} \sum_{i=1}^n X_i X_i' \right)^{-1}$$

and then we can construct a 95% confidence interval by

$$\hat C = \left[\hat\theta \pm 1.96 \frac{\sqrt{\hat V}}{\sqrt{n}} \right]$$

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 $\sqrt{n}(\hat\theta - \theta)$. Using a delta method type of argument, notice that we have that

$$r(\hat\beta_1, \hat\beta_2) = r(\beta_1, \beta_2) + \nabla r(\bar\beta_1, \bar\beta_2)' \begin{pmatrix} \hat\beta_1 - \beta_1 \\ \hat\beta_2 - \beta_2 \end{pmatrix} \tag{2}$$

where

$$\begin{aligned}\nabla r(\bar{b}_1, \bar{b}_2) &= \begin{bmatrix} \frac{\partial r(b_1,b_2)}{\partial b_1} \\ \\ \frac{\partial r(b_1,b_2)}{\partial b_2} \end{bmatrix}\Bigg|_{b_1=\bar{b}_1, b_2=\bar{b}_2} \\ &= \begin{bmatrix} 1 \\ \\ -1 \end{bmatrix}\end{aligned}$$

In other words, $\nabla r(\bar{b}_1, \bar{b}_2)$ is the vector of partial derivatives of r with respect to each of its arguments evaluated at $\bar{b}_1$ and $\bar{b}_2$. For the particular r in our problem, the vector of partial derivatives is just equal to $(1, -1)'$ no matter the values of $\bar{b}_1$ and $\bar{b}_2$. Plugging this back into Equation 2 and multiplying by $\sqrt{n}$, we have that

$$\begin{aligned}\sqrt{n}(\hat{\theta} - \theta) &= \begin{bmatrix} 1 \\ \\ -1 \end{bmatrix}' \sqrt{n} \begin{pmatrix} \hat{\beta}_1 - \beta_1 \\ \hat{\beta}_2 - \beta_2 \end{pmatrix} \\ &\xrightarrow{d} N(0, V)\end{aligned}$$

(as a small side-comment, in the first line there is no $o_p(1)$ term at the end of that equation — that line holds exactly from the previous part because the vector of partial derivatives of r does not depend on the values of $\bar{b}_1$ and $\bar{b}_2$) and where

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

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 2×2variance matrix, like $\mathbf{V}_\beta$ as

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

As a side-comment, this 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 time 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

$$\begin{aligned}\hat{C}^{wide} &= \left[\hat{\theta} \pm 1.96\sqrt{\text{se}(\hat{\beta}_1)^2 + 2\text{se}(\hat{\beta}_1)\text{se}(\hat{\beta}_2) + \text{se}(\hat{\beta}_2)^2}\right] \\ &= \left[0.2 \pm 1.96\sqrt{3 \times 0.07^2}\right] \\ &= [-0.038, 0.438]\end{aligned}$$

This includes 0, which suggests that the author's claim is not correct; the difference between $\hat{\beta}_1$ and $\hat{\beta}_2$ is (just barely) not statistically significant.

7.28

(a)

I am going to include a little bit of extra detail about comparing "manually" calculated standard errors with those coming directly from R as I think this is interesting. For part of the problem, I'll compare to results from the R package `estimatr` which is popular among economists for computing heteroskedasticity robust standard errors.

```
# read data
library(haven)
```

```r
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$wage)
X <- cbind(1, data$education, data$exp, data$exp^2/100)
bet <- solve(t(X)%*%X)%*%t(X)%*%Y
round(bet,3)
```

```
##        [,1]
## [1,]  1.185
## [2,]  0.090
## [3,]  0.035
## [4,] -0.047
```

```r
# 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),3)
```

```
##     beta    se
## 1  1.185 0.046
## 2  0.090 0.003
## 3  0.035 0.003
## 4 -0.047 0.005
```

```r
# compare to R's lm function
reg <- lm(log(wage) ~ education + exp + I(exp^2/100), data=data)
summary(reg)
```

```
##
## Call:
## lm(formula = log(wage) ~ education + exp + I(exp^2/100), data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.0275 -0.3135  0.0063  0.3411  2.8603
##
```

```
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.185209   0.044745  26.488   <2e-16 ***
## education      0.090449   0.002737  33.051   <2e-16 ***
## exp            0.035380   0.002512  14.083   <2e-16 ***
## I(exp^2/100) -0.046506   0.005027  -9.251   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5739 on 4226 degrees of freedom
## Multiple R-squared:  0.2334, Adjusted R-squared:  0.2328
## F-statistic: 428.8 on 3 and 4226 DF,  p-value: < 2.2e-16
```

```
# Notice that estimates of beta are the same but
# standard errors are different

library(estimatr)
reg2 <- lm_robust(log(wage) ~ education + exp + I(exp^2/100), data=data, se_type="HC0")
summary(reg2)
```

```
##
## Call:
## lm_robust(formula = log(wage) ~ education + exp + I(exp^2/100),
##     data = data, se_type = "HC0")
##
## Standard error type:  HC0
##
## Coefficients:
##              Estimate Std. Error t value  Pr(>|t|) CI Lower CI Upper   DF
## (Intercept)   1.18521   0.046078  25.722 1.129e-135  1.09487  1.27555 4226
## education     0.09045   0.002915  31.028 1.312e-190  0.08473  0.09616 4226
## exp           0.03538   0.002584  13.691  8.859e-42  0.03031  0.04045 4226
## I(exp^2/100) -0.04651   0.005304  -8.767  2.606e-18 -0.05691 -0.03611 4226
##
## Multiple R-squared:  0.2334 ,    Adjusted R-squared:  0.2328
## F-statistic: 372.7 on 3 and 4226 DF,  p-value: < 2.2e-16
```

```
# these are exactly the same now

# Homoskedasticity standard errors
sigma2 <- mean(ehat^2)
V0 <- sigma2 * solve(XX)
se0 <- sqrt(diag(V0))/sqrt(n)
se0
```

```
## [1] 0.044723948 0.002735371 0.002511055 0.005024746
```

```
# these are very, very close to R's lm standard errors
# but not exactly the same
```

```r
# Homoskedasticity w/ degree of freedom adjustment
k <- 4 # number of regressors (including intercept)
s2 <- sum(ehat^2)/(n-k)
Vs <- s2 * solve(XX)
ses <- sqrt(diag(Vs))/sqrt(n)
ses
```

```
## [1] 0.044745109 0.002736665 0.002512243 0.005027124
```

```r
# these are exactly the same now
```

(e)

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

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

```r
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

```r
# 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
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
```

```r
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 2

(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))
```

```r
}

# 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)

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

```
## rej. prob  :  0.256
```

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

```
## rej. prob  :  0.1
```

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

```
## rej. prob  :  0.052
```

```r
# 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 $\sqrt{n}(\hat{\beta} - \beta)$ 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.

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

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

```
## rej. prob  :  0.439
```

```r
# 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.