HW5

Sandra Villamar and Shobhit Dronamraju

2023-02-16

Problem 1

Write a function named bootLS(x, y, conf = 0.95, B = 1000) that fits a simple linear model explaining y in terms of x, and returns a studentized bootstrap confidence interval at the desired level based on the specified number of repeats for each coefficient vector.

```r
bootLS <- function(x, y, conf = 0.95, B = 1000){

  # fit linear model to original sample
  fit = lm(y ~ x)
  beta0 = fit$coefficients[1]
  beta1 = fit$coefficients[2]
  sebeta0=summary(fit)$coefficients[,2][1]
  sebeta1=summary(fit)$coefficients[,2][2]

  # set up variables to store
  N = length(x)
  beta0_boot = rep(NA,N)
  beta1_boot = rep(NA,N)
  t0_boot = rep(NA,N)
  t1_boot = rep(NA,N)

  # generate bootstrap samples and stats
  for (i in 1:B){
    indices = sample(1:N, N, replace=TRUE)
    x_boot = x[indices]
    y_boot = y[indices]
    fit_boot = lm(y_boot ~ x_boot)
    beta0_boot[i] = fit_boot$coefficients[1]
    beta1_boot[i] = fit_boot$coefficients[2]
    sebeta0_boot = summary(fit_boot)$coefficients[,2][1]
    sebeta1_boot = summary(fit_boot)$coefficients[,2][2]
    t0_boot[i] = (beta0_boot[i] - beta0) / (sebeta0_boot)
    t1_boot[i] = (beta1_boot[i] - beta1) / (sebeta1_boot)
  }

  # CI for intercept
  boot_int = matrix(c(beta0
                      + quantile(t0_boot, c((1 - conf) / 2, (1 + conf) / 2))
                      * sebeta0),
                    ncol = 2)
  colnames(boot_int) = c('2.5 %', '97.5 %')
```

```
  # CI for slope
  boot_slp = matrix(c(beta1
                      + quantile(t1_boot, c((1 - conf) / 2, (1 + conf) / 2))
                      * sebeta1),
                    ncol = 2)
  colnames(boot_slp) = c('2.5 %', '97.5 %')

  return(rbind(boot_int, boot_slp))
}
```

```
# testing function
n = 200
x = runif(n, -1, 1)
y = 1 + 2*x + rnorm(n, 0, 0.5)

bootLS(x, y)
```

```
##          2.5 %   97.5 %
## [1,] 0.9426448 1.074816
## [2,] 1.9049222 2.149798
```

```
fit = lm(y~x)
confint(fit)
```

```
##                 2.5 %   97.5 %
## (Intercept) 0.9430344 1.078211
## x           1.9020614 2.149561
```

Problem 2

Perform some simulations to compare the length and confidence level of the studentized bootstrap confidence interval (from Problem 1) and of the student confidence interval (the classical one). Compare them at various sample sizes and in settings involving different distributions, for example, the normal distribution and a skewed distribution like the exponential distribution (centered to have mean 0). In the code, first briefly explain in words what you intend to do, and then do it, and at the end offer some brief comments on the results of your simulation study.

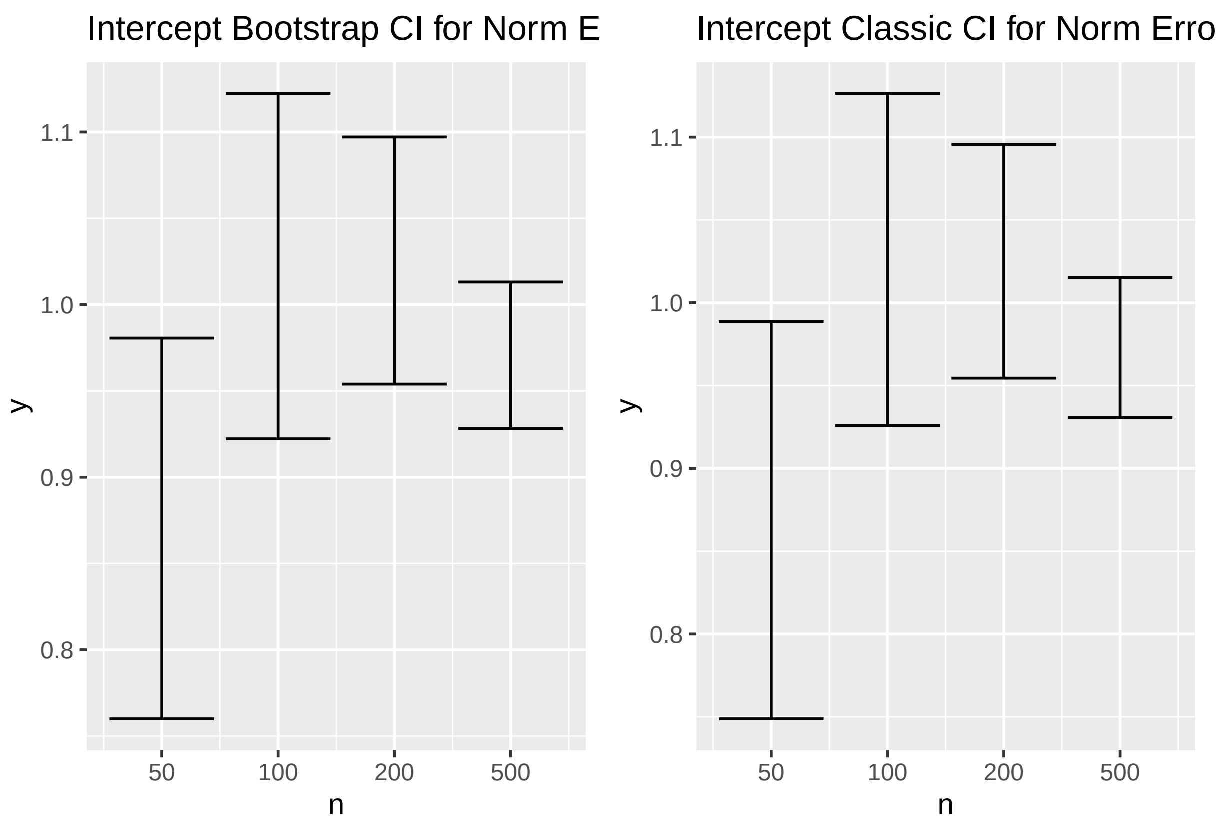
Simulation Plan: We will compare the studentized bootstrap interval (Problem 1) and student confidence interval (classical) with the following settings:

- sample sizes: n = {50, 100, 200, 500}
- errors have normal distribution: $y \sim N(1 + 2x, \sigma^2)$ s.t. $x \sim U(-1, 1)$ and $\sigma = 0.5$
- errors have skewed distribution: $y \sim 1 + 2x + [Exp(\lambda) - \frac{1}{\lambda}]$ s.t. $x \sim U(-1, 1)$ and $\lambda = 0.5$

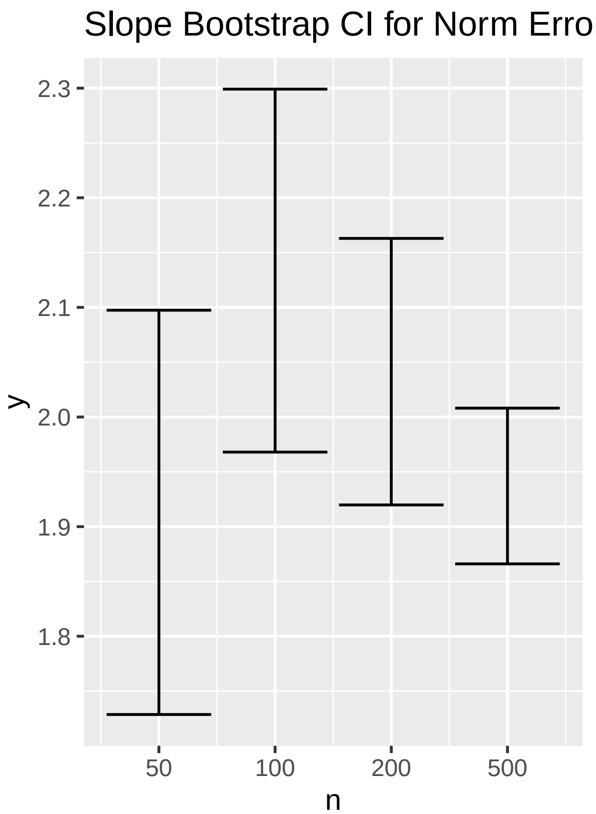
```
ns = c(50, 100, 200, 500)  # sample sizes

intervals_int = data.frame(n = 1:length(ns),
                           lower_boot = rep(NA, length(ns)),
                           upper_boot = rep(NA, length(ns)),
                           lower_clas = rep(NA, length(ns)),
```

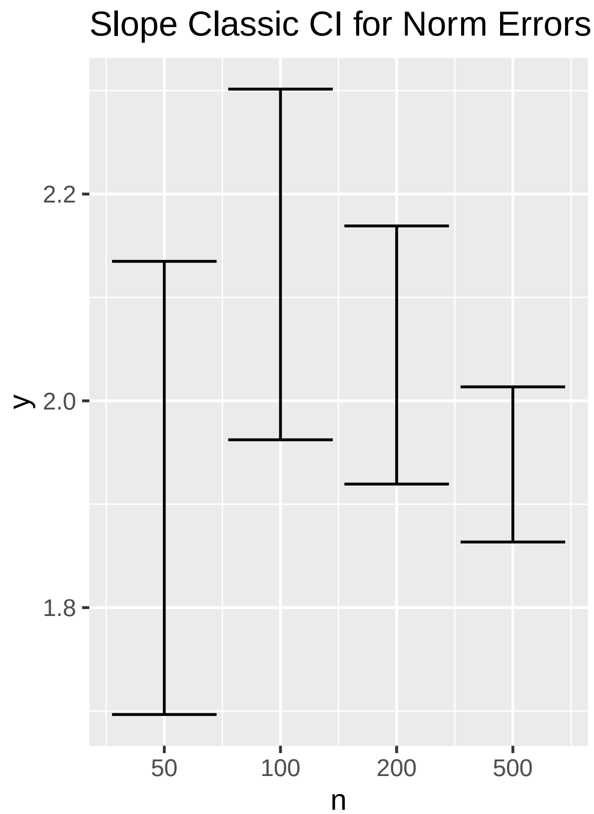
```
                            upper_clas = rep(NA, length(ns)))  # store CIs
intervals_slope = data.frame(n = 1:length(ns),
                              lower_boot = rep(NA, length(ns)),
                              upper_boot = rep(NA, length(ns)),
                              lower_clas = rep(NA, length(ns)),
                              upper_clas = rep(NA, length(ns)))  # store CIs
# normalized errors
for(i in 1:length(ns)){
  n = ns[i]
  x = runif(n, -1, 1)
  y = 1 + 2*x + rnorm(n, 0, 0.5)

  confint_boot = bootLS(x, y)
  fit = lm(y~x)
  confint_classic = confint(fit)

  intervals_int[i, 2:3] = confint_boot[1,]
  intervals_slope[i, 2:3] = confint_boot[2,]
  intervals_int[i, 4:5] = confint_classic[1,]
  intervals_slope[i, 4:5] = confint_classic[2,]
}
```

```
# plots to compare
library("ggplot2")
library("patchwork")

ticks = paste(ns)

# intercept CI for normalized errors
p1 = ggplot(intervals_int, aes(n, lower_boot)) +
      geom_errorbar(aes(ymin = lower_boot, ymax = upper_boot))
p1 = p1 + scale_x_continuous(breaks=intervals_int$n, labels=ticks) +
        ylab("y") + labs(title = "Intercept Bootstrap CI for Norm Errors")

p2 = ggplot(intervals_int, aes(n, lower_clas)) +
      geom_errorbar(aes(ymin = lower_clas, ymax = upper_clas))
p2 = p2 + scale_x_continuous(breaks=intervals_int$n, labels=ticks) +
        ylab("y") + labs(title = "Intercept Classic CI for Norm Errors")

p1 + p2
```




```r
# slope CI for normalized errors
p3 = ggplot(intervals_slope, aes(n, lower_boot)) +
      geom_errorbar(aes(ymin = lower_boot, ymax = upper_boot))
p3 = p3 + scale_x_continuous(breaks=intervals_slope$n, labels=ticks) +
        ylab("y") + labs(title = "Slope Bootstrap CI for Norm Errors")

p4 = ggplot(intervals_slope, aes(n, lower_clas)) +
      geom_errorbar(aes(ymin = lower_clas, ymax = upper_clas))
p4 = p4 + scale_x_continuous(breaks=intervals_slope$n, labels=ticks) +
        ylab("y") + labs(title = "Slope Classic CI for Norm Errors")

p3 + p4
```



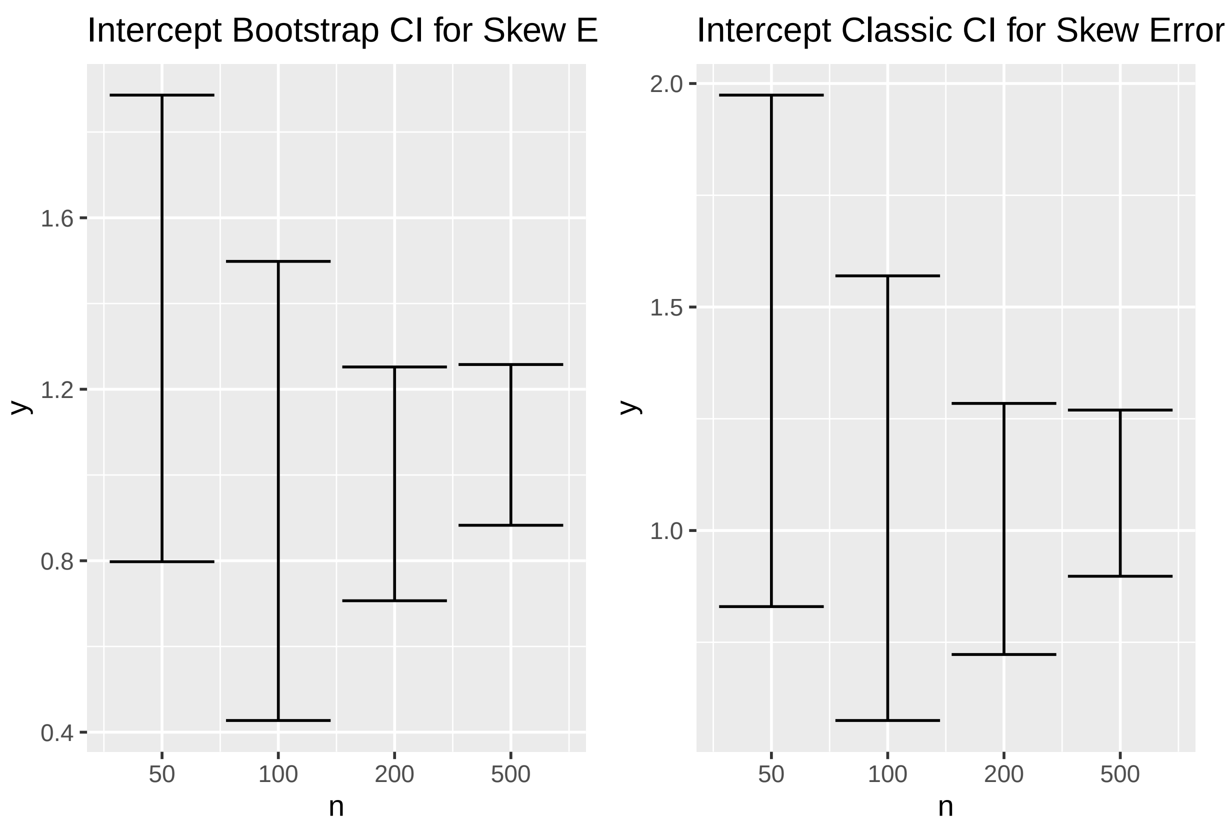


We see that as the sample size increases, the confidence intervals (bootstrap and classical) become narrower. This makes sense, as the more data we have in our sample, the better representation of the actual distribution we have. There is no apparent difference between the bootstrap and classical intervals, the upper and lower bounds are all quite close to each other.

```
intervals_int2 = data.frame(n = 1:length(ns),
                            lower_boot = rep(NA, length(ns)),
                            upper_boot = rep(NA, length(ns)),
                            lower_clas = rep(NA, length(ns)),
                            upper_clas = rep(NA, length(ns)))  # store CIs
intervals_slope2 = data.frame(n = 1:length(ns),
                              lower_boot = rep(NA, length(ns)),
                              upper_boot = rep(NA, length(ns)),
                              lower_clas = rep(NA, length(ns)),
                              upper_clas = rep(NA, length(ns)))  # store CIs
# skewed errors
for(i in 1:length(ns)){
  n = ns[i]
  x = runif(n, -1, 1)
  y = 1 + 2*x + rexp(n, 0.5) - 2

  confint_boot = bootLS(x, y)
  fit = lm(y~x)
  confint_classic = confint(fit)

  intervals_int2[i, 2:3] = confint_boot[1,]
  intervals_slope2[i, 2:3] = confint_boot[2,]
```

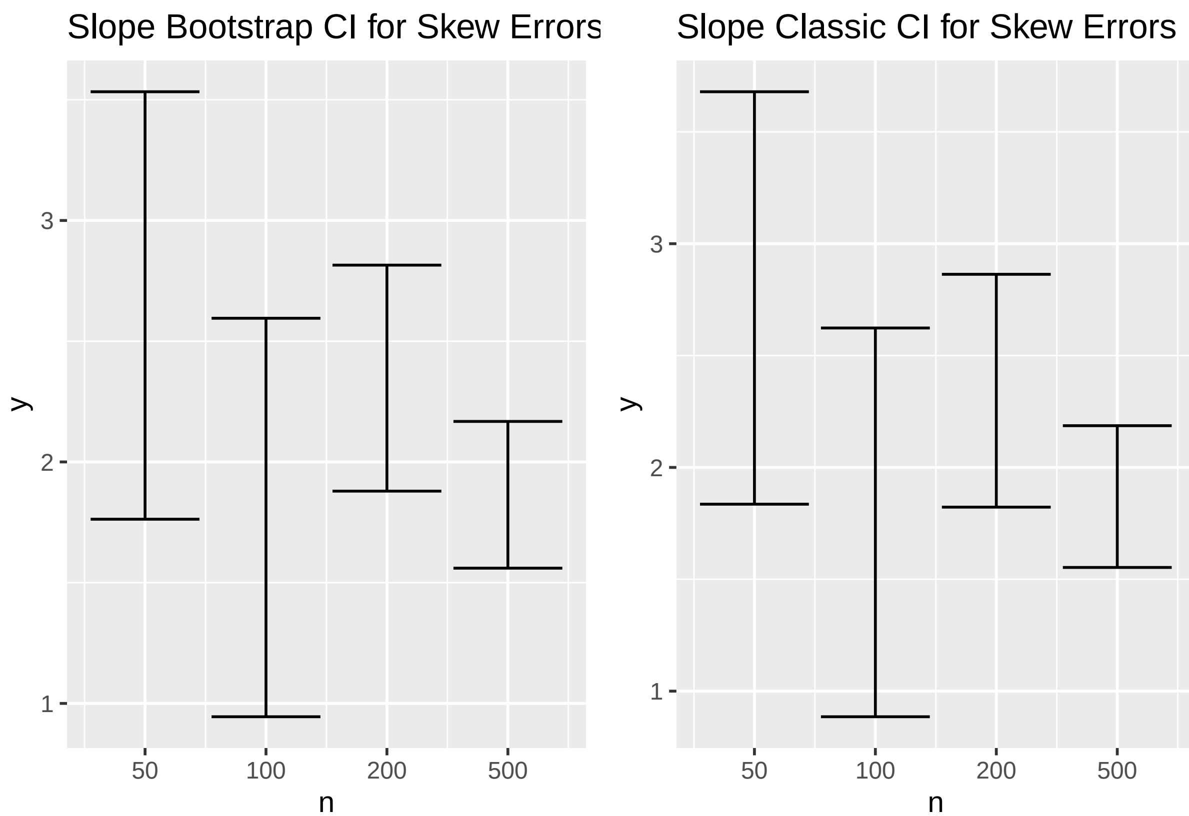
```
  intervals_int2[i, 4:5] = confint_classic[1,]
  intervals_slope2[i, 4:5] = confint_classic[2,]
}
```

```
# plots to compare

# intercept CI for skewed errors
p1 = ggplot(intervals_int2, aes(n, lower_boot)) +
      geom_errorbar(aes(ymin = lower_boot, ymax = upper_boot))
p1 = p1 + scale_x_continuous(breaks=intervals_int2$n, labels=ticks) +
        ylab("y") + labs(title = "Intercept Bootstrap CI for Skew Errors")

p2 = ggplot(intervals_int2, aes(n, lower_clas)) +
      geom_errorbar(aes(ymin = lower_clas, ymax = upper_clas))
p2 = p2 + scale_x_continuous(breaks=intervals_int2$n, labels=ticks) +
        ylab("y") + labs(title = "Intercept Classic CI for Skew Errors")

p1 + p2
```




```
# slope CI for skewed errors
p3 = ggplot(intervals_slope2, aes(n, lower_boot)) +
      geom_errorbar(aes(ymin = lower_boot, ymax = upper_boot))
p3 = p3 + scale_x_continuous(breaks=intervals_slope2$n, labels=ticks) +
        ylab("y") + labs(title = "Slope Bootstrap CI for Skew Errors")
```

```
p4 = ggplot(intervals_slope2, aes(n, lower_clas)) +
      geom_errorbar(aes(ymin = lower_clas, ymax = upper_clas))
p4 = p4 + scale_x_continuous(breaks=intervals_slope2$n, labels=ticks) +
        ylab("y") + labs(title = "Slope Classic CI for Skew Errors")

p3 + p4
```



Again we see that as the sample size increases, the confidence intervals (bootstrap and classical) become narrower. This makes sense, as the more data we have in our sample, the better representation of the actual distribution we have. There is no apparent difference between the bootstrap and classical intervals, the upper and lower bounds are all quite close to each other.

We know that the *actual* distribution has intercept 1 and slope 2. For both the normalized and skewed errors, the true parameter values are captured in the confidence intervals most of the time.