# ps3

Sijun Zhang 89934761

2019/11/5

# Question 1

## Part a

The result is shown in a 1x2 matrix and the left element shows the lower bound and the right element shows the upper bound:

```r
# Part a -- Jacknife Method
jacknife_ci = function(x, y, alpha = 0.05) {
  nx = length(x)
  ny = length(y)
  x_s = sum(x)
  y_s = sum(y)
  theta_i_c_x = ((x_s - x) / (nx - 1)) / mean(y)
  theta_i_c_y = mean(x) / ((y_s - y) / (ny - 1))
  theta_i_c = append(theta_i_c_x, theta_i_c_y)
  sigma = sqrt((nx + ny -1) / (nx + ny) * sum( (theta_i_c - mean(theta_i_c))^2) )
  return( list(c(mean(x)/mean(y) + qnorm(alpha/2)*sigma,
           mean(x)/mean(y) + qnorm(1-alpha/2)*sigma)) )
}

x = matrix(1:4, nrow = 2, byrow = TRUE)
y = matrix(1:6, nrow = 2, byrow = TRUE)
jacknife_ci(x[1,],y[1,])
```

```
## [[1]]
## [1] -0.04951157  1.54951157
```

## Part b

The result is shown in a 1x6 matrix and the columns 1,3,5 shows the lower bounds and the columns 2,4,6 shows the upper bounds, the columns 1,2 shows the confidence interval from percentile method, the columns 3,4 shows the confidence interval from basic bootstrap and the columns 5,6 shows the confidence interval from normal approximation:

```r
# Part b -- Bootstrap
boot_ci = function(x, y, nboot = 1e4, alpha = 0.05) {
  # method could take "percentile method", "basic bootstrap" and
  # "normal approximation"
  nx = length(x)
  ny = length(y)
  x_m = sample(x, nboot * nx, replace = TRUE)
  y_m = sample(y, nboot * ny, replace = TRUE)
  dim(x_m) = c(nboot, nx)
  dim(y_m) = c(nboot, ny)
  x_m_mean = rowMeans(x_m)
  y_m_mean = rowMeans(y_m)
  theta_i_star = x_m_mean / y_m_mean
  sigma = sqrt(var(theta_i_star))
  # percentile method and basic bootstrap
  re = append(quantile(theta_i_star, c(alpha/2, 1-alpha/2)),
              2*mean(x)/mean(y) - quantile(theta_i_star, c(1-alpha/2, alpha/2)))
  # normal approximation
  re = append(re, c(mean(x)/mean(y)-qnorm(1-alpha/2)*sigma) )
  re = append(re, c(mean(x)/mean(y)+qnorm(1-alpha/2)*sigma) )
  return(list(unname(re)))
}
boot_ci(x[1,],y[1,])
```

```
## [[1]]
## [1] 0.3750000 1.5000000 0.0000000 1.1250000 0.1755181 1.3244819
```

## Part c

```r
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------------------------- tidyverse 1.2.1 -
-
```

```
## √ ggplot2 3.2.1     √ purrr   0.3.2
## √ tibble  2.1.3     √ dplyr   0.8.3
## √ tidyr   1.0.0     √ stringr 1.4.0
## √ readr   1.3.1     √ forcats 0.4.0
```

```
## -- Conflicts ---------------------------------------------------------------------- tidyverse_conflicts() -
-
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
data("ToothGrowth")
TG = tibble::as_tibble(ToothGrowth)
CI_TG = TG %>%
  # select(len, supp) %>%
  group_by(dose) %>%
  pivot_wider(values_from = len, names_from = supp) %>%
  summarize(jacknife_ci = jacknife_ci(unlist(OJ), unlist(VC)),
            bootstrap_ci = boot_ci(unlist(OJ), unlist(VC)),
            point_est = mean(unlist(OJ)) / mean(unlist(VC))) %>%
  group_by(dose) %>%
  mutate(jacknife_lcb = unlist(jacknife_ci)[1],
         jacknife_ucb = unlist(jacknife_ci)[2],
         percentile_lcb = unlist(bootstrap_ci)[1],
         percentile_ucb = unlist(bootstrap_ci)[2],
         basic_bs_lcb = unlist(bootstrap_ci)[3],
         basic_bs_ucb = unlist(bootstrap_ci)[4],
         normal_lcb = unlist(bootstrap_ci)[5],
         normal_ucb = unlist(bootstrap_ci)[6]) %>%
  select(-jacknife_ci, -bootstrap_ci)
knitr::kable(CI_TG)
```

| dose | point_est | jacknife_lcb | jacknife_ucb | percentile_lcb | percentile_ucb | basic_bs_lcb | basic_bs_ucb | normal_lcb | normal_ucb |
|---:|---:|---:|---:|---:|---:|---:|---:|---:|---:|
| 0.5 | 1.6578947 | 1.1471504 | 2.168639 | 1.2460660 | 2.212744 | 1.1030457 | 2.069723 | 1.1747252 | 2.141064 |
| 1.0 | 1.3536076 | 1.1551440 | 1.552071 | 1.1731618 | 1.539242 | 1.1679735 | 1.534053 | 1.1719378 | 1.535278 |
| 2.0 | 0.9969396 | 0.8631894 | 1.130690 | 0.8848095 | 1.129474 | 0.8644055 | 1.109070 | 0.8739134 | 1.119966 |

The confidence intervals and the point estimates for each level are shown above.

# Question 2

In this question, if we have ntot samples and nx observations within each x sample, ny observations within each y samples, when dim(x) = c(ntot, nx), dim(y) = c(ntot, ny), the code will be efficient (row represents samples).

## Part a

The result is shown in a ntotx2 matrix and the left column shows the lower bounds and the right column shows the upper bounds:

```r
# Part a -- Jacknife for MC
mc_jacknife_ci = function(x, y, alpha = 0.05) {
  # if we have ntot samples and nx observations within each x sample,
  # ny observations within each y samples, when dim(x) = c(ntot, nx),
  # dim(y) = c(ntot, ny), the code will be efficient (row represents samples).
  ntot = dim(x)[1]
  nx = dim(x)[2]
  ny = dim(y)[2]
  x_s = rowSums(x)
  y_s = rowSums(y)
  x_m = x_s - x
  y_m = y_s - y
  theta_i_c_x = (x_m / (nx - 1)) / rowMeans(y)
  theta_i_c_y = rowMeans(x) / (y_m / (ny - 1))
  theta_i_c = cbind(theta_i_c_x, theta_i_c_y)
  sums = rowSums((theta_i_c - rowMeans(theta_i_c))^2)
  sigma = sqrt((nx + ny -1) / (nx + ny) * sums)
  return(cbind(matrix(rowMeans(x)/rowMeans(y) + qnorm(alpha/2)*sigma, ncol = 1),
         matrix(rowMeans(x)/rowMeans(y) + qnorm(1-alpha/2)*sigma, ncol = 1) ) )
}

x = matrix(1:4, nrow = 2, byrow = TRUE)
y = matrix(1:6, nrow = 2, byrow = TRUE)
mc_jacknife_ci(x,y)
```

```
##              [,1]     [,2]
## [1,] -0.04951157 1.549512
## [2,]  0.39606539 1.003935
```

## Part b

The result is shown in a ntotx6 matrix and the columns 1,3,5 shows the lower bounds and the columns 2,4,6 shows the upper bounds, the columns 1,2 shows the confidence interval from percentile method, the columns 3,4 shows the confidence interval from basic bootstrap and the columns 5,6 shows the confidence interval from normal approximation:

```r
# Part b -- Bootstrap for MC
mc_bootstrap_ci = function(x, y, nboot = 1e4, alpha = 0.05) {
  ntot = dim(x)[1]
  nx = dim(x)[2]
  ny = dim(y)[2]
  # Vectorized sample using index reference
  index_x = sample(1:nx, nboot * nx * ntot, replace = TRUE)
  index_y = sample(1:ny, nboot * ny * ntot, replace = TRUE)
  dim(index_x) = c(nx * nboot, ntot)
  index_shifter_x = matrix(1, ncol = ntot, nrow = nx * nboot) %*% diag(0:(ntot-1)) * nx
  index_x = as.numeric(index_x + index_shifter_x)
  x_m = matrix(t(x)[index_x], byrow = TRUE, nrow = ntot * nboot)
  dim(index_y) = c(ny * nboot, ntot)
  index_shifter_y = matrix(1, ncol = ntot, nrow = ny * nboot) %*% diag(0:(ntot-1)) * ny
  index_y = as.numeric(index_y + index_shifter_y)
  y_m = matrix(t(y)[index_y], byrow = TRUE, nrow = ntot * nboot)
  # calc theta bootstrap sets and normal approx standard error
  x_m_mean = matrix(rowMeans(x_m), byrow = TRUE, nrow = ntot)
  y_m_mean = matrix(rowMeans(y_m), byrow = TRUE, nrow = ntot)
  theta_i_star = x_m_mean / y_m_mean
  sigma = diag(sqrt(var(t(theta_i_star))))
  # percentile method
  re1 = unname(t(apply(theta_i_star, 1, quantile, c(alpha/2, 1-alpha/2) )) )
  # basic bootstrap
  re2 = cbind(matrix((2*rowMeans(x)/rowMeans(y) - re1)[,2], ncol = 1),
             matrix((2*rowMeans(x)/rowMeans(y) - re1)[,1], ncol = 1))
  # normal approx
  re3 = cbind(matrix(rowMeans(x)/rowMeans(y)-sigma*qnorm(1-alpha/2), ncol = 1),
              matrix(rowMeans(x)/rowMeans(y)-sigma*qnorm(alpha/2), ncol = 1))
  # combine result
  re = cbind(re1, re2)
  re = cbind(re, re3)
  return(re)
}
mc_bootstrap_ci(x,y)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 0.3750000 1.5000000 0.0000000 1.1250000 0.1785254 1.3214746
## [2,] 0.5294118 0.9230769 0.4769231 0.8705882 0.5041757 0.8958243
```

## Part c

In this part, we assume each x_i and y_i all follow normal distributions and the expaction of x is e_x = 3, the expectation of y is e_y = 4, and their variance is 2 and 3, respectively, we generated ntot = 1e2 mc samples to find the ratio theta = E[X] / E[Y], when calculating the confidence interval, we use the default confidence level alpha = 0.05 and use nboot = 10000 to generate bootstrap samples.

```r
# Part c -- Verification

# In this part, we assume each x_i and y_i all follow normal distributions and the
# expaction of x is e_x = 3, the expectation of y is e_y = 4, and their variance
# is 2 and 3, respectively, we generated ntot = 1e2 mc samples to find the ratio
# theta = E[X] / E[Y], when calculating the confidence interval, we use the default
# confidence level alpha = 0.05 and use nboot = 10000 to generate bootstrap samples.

nx = 10
ny = 20
e_x = 3
e_y = 4
theta = e_x / e_y
ntot = 1e2
x = rnorm(nx*ntot, mean = e_x, sd = 2)
dim(x) = c(ntot, nx)
y = rnorm(ny*ntot, mean = e_y, sd = 3)
dim(y) = c(ntot, ny)

jk_ci = mc_jacknife_ci(x,y)
set.seed(196)
bs_ci = mc_bootstrap_ci(x,y)
ci = cbind(jk_ci, bs_ci)

# The coverage probability, the true value of theta is 3/4
cov_prob = cbind(cbind(sum((theta >= ci[,1]) & (theta <= ci[,2])),
                       sum((theta >= ci[,3]) & (theta <= ci[,4]))),
                 cbind(sum((theta >= ci[,5]) & (theta <= ci[,6])),
                       sum((theta >= ci[,7]) & (theta <= ci[,8]))))/ntot
# The average length of confidence interval
avg_len = cbind(cbind(mean(ci[,2] - ci[,1]),
                      mean(ci[,4] - ci[,3])),
                cbind(mean(ci[,6] - ci[,5]),
                      mean(ci[,8] - ci[,7])))
# The average shape of the confidence intervals produced by each method,
point_est = rowMeans(x) / rowMeans(y)
len_ratio = cbind(cbind(mean((ci[,2] - point_est)/(point_est - ci[,1])),
                        mean((ci[,4] - point_est)/(point_est - ci[,3]))),
                  cbind(mean((ci[,6] - point_est)/(point_est - ci[,5])),
                        mean((ci[,8] - point_est)/(point_est - ci[,7]))))
quantities = rbind(rbind(cov_prob,
                         avg_len),
                   len_ratio)
quantities = as.data.frame(quantities, row.names = c("cov_prob",
                                                     "avg_len",
                                                     "len_ratio"))
names(quantities) = c("Jacknife", "Percentile_bootstrap",
                      "Basic_bootstrap", "Normal_approx_bootstrap")
knitr::kable(quantities)
```

| | Jacknife | Percentile_bootstrap | Basic_bootstrap | Normal_approx_bootstrap |
|---|---|---|---|---|
| cov_prob | 0.9600000 | 0.9400000 | 0.9500000 | 0.960000 |
| avg_len | 0.9025713 | 0.9455322 | 0.9455322 | 7.076442 |
| len_ratio | 1.0000000 | 1.6491720 | 0.6382090 | 1.000000 |

From the quantities, under the assumption that x and y both follow normal distribution, we can see that percentile method using bootstarp come up with the least coverage probability, the Jacknife and normal approximation acheive the highest coverage probability.

Moreover, the normal approximation in bootstrap has the largest average length in confidence interval, the lengths of percentil and basic bootstrap confidence interval are the same.

Jacknife confidence interval and Normal approximation confidence interval are both symmetric confidence interval, both the percentile one and the basic one has the asymmetric confidence interval with a approximate difference 0.5 between the left half and right. The percentile confidence interval has a larger distance between the point estimate the upperbound, and the basic confidence interval has a larger distance between the point estimate the lowerbound.