Homework Assignment #2

Math 437 - Modern Data Analysis

Due February 24, 2023

Instructions

You should submit either two or three files:

1. You should write your solutions to the Simulation and Applied Problems in this R Markdown file and submit the (.Rmd) file.
2. You should knit the final solution file to pdf and submit the pdf. If you are having trouble getting code chunks to run, add `eval = FALSE` to the chunks that do not run. If you are having trouble getting R Studio to play nice with your LaTeX distribution, I will begrudgingly accept an HTML file instead.
3. Solutions to the Key Terms and Conceptual Problems can be submitted in a separate Word or pdf file or included in the same files as your solutions to the Simulation and Applied Problems.

This homework assignment is worth a total of **40 points**.

Key Terms (5 pts)

Read Chapter 13 of Introduction to Statistical Learning, Second Edition. Based on your reading, answer the following questions.

1. What is a *p-value*? What is the difference between a one-sided and a two-sided p-value?

Answer: The p-value is the probability of observing a test statistic that is equal to or more extreme than the observed statistic under the assumption that H_0 is true. The difference between the one and two sided p-value is a two sided p-value is based on the absolute value of the test statistic, i.e. our t-score could be 1.65 or -1.65 but the one sided p-value is only looking at one of those values.

2. In traditional NHST-style significance testing, what are the two possible decisions? When do we make each decision?

Answer: The two possible decisions are to reject the null hypothesis, occurs when our p-value is less than our pre-determined alpha. The other is to fail to reject our null hypothesis which occurs when our p-value is larger than that alpha.

3. What is the difference between a *Type I Error* and a *Type II Error*?

Answer: The best case of Type I and Type II error I have learned is the story of the boy who cried wolf. At first the city makes a Type I error by rejecting H_0 (there is no wolf) when in fact there was no wolf, followed by them making a Type II error by not rejecting H_0 (again there is no wolf) when in reality there is. So a Type I Error occurs when we reject H_0 when H_0 was in fact true and a Type II Error is occurs when we do not reject H_0 when H_a was true.

4. Briefly explain why it is necessary to adjust the significance level (or equivalently, the p-values) when testing a large number of null hypotheses. Answer: The reason why we need to adjust the significance level when testing a large number of null hypotheses is to account for the chance of seeing a small p-value strictly by chance. When a large number of tests are conducted, the number of Type I Errors

we expect to make will increase so it becomes necessary to adjust the significance level to control for this.

5. Compare and contrast the *Family-Wise Error Rate* (FWER) and the *False Discovery Rate* (FDR). Answer: The FWER and FDR are similar in the sense that both are looking at the likelihood of Type I Error rates are but are very different in ideas. FWER looks at the probability of making at least one Type I Error across m tests whereas FDR is looking to minimize the proportion of type one error rates and total rejections of H_0.

6. Compare and contrast the *Bonferroni Method* and *Holm's Step-Down Method* for controlling the FWER. Answer: Bonferroni & Holm's Step-Down Method both aim to decrease (or control) the FWER based on the number of tests conducted without making assumptions about the form of H_0, the choice of test statistic, or the (in)dependence of p-values. Bonferroni's method adjusts the significance level based purely on the number of tests conducted, m, so that our new significance level is $\alpha^* = \frac{\alpha}{m}$. Holm's Step-Down Method ranks the p-values from every test in ascending order and begins comparing them beginning with the most conservative significance level of $\alpha^* = \frac{\alpha}{m}$ for the smallest p-value and if the p-value is significant, it adjusts the significance level for the subsequent p-value to be $\alpha^* = \frac{\alpha}{m-1}$ and so on until a non-significant p-value is found, resulting in a higher number of null hypotheses rejected.

7. Why do we prefer to use *Tukey's Method* or *Scheffe's Method* to control the FWER? In what conditions is it appropriate to use those methods instead of the Bonferroni or Holm methods?

Answer: We prefer to use Tukey's or Scheffe's Method because they allows us to further control the FWER while increasing power. It is more appropriate to use Tukeys methods when we have dependence between p-values caused by similar hypotheses tests so we can conduct pairwise comparisons and we can use Scheffe's to control FWER at a specific significance level α.

8. Briefly describe the *Benjamini-Hochberg* procedure for controlling the FDR. Answer: The Benjamini-Hochberg procedure begins by deciding on a specific q value to control the FDR at. Then finding and ordering our p-values such that the smallest p-value is first to the largest p-value. Next we define L as the maximum p-value for which the jth p-value is less than q(j/m). Lastly we reject all p-values for which p(j) is less than p(L).

9. What is/are the major assumption(s) of a *permutation test*? What is the general procedure for obtaining the null distribution of a test statistic using a permutation test? Answer: The major assumptions of a permutation test is that the distribution of X and Y are similar such that values can be interchanged when calculating our T-values. The process of obtaining the null distribution is through permuting n(x) + n(y) B times for a large number, B. Once that is done the distribution of B is that null distribution.

10. When is it useful/recommended to use a permutation testing approach as opposed to a traditional theory-based approach? Answer: We would want to use a permutation test when the distribution of our data does not follow a known shape. When our distribution follows that of a t-distribution, chi-squared distribution, F-distribution, etc. a traditional based approach works wonderful but when we are not following that we are inclined to go towards permutation methods.

Conceptual Problems

Conceptual Problem 1 (5 pts)

Textbook Exercise 13.7.6:

For each of the three panels in Figure 13.3, answer the following questions: (a) How many false positives, false negatives, true positives, true negatives, Type I errors, and Type II errors result from applying the Bonferroni procedure to control the FWER at level $\alpha = 0.05$?

1. Panel one has seven true positives, one false negative, and two true negatives. Thus it has One Type II Error and no Type I Errors.

2. Panel two has seven true positives, one false negative, and two true negatives. Thus it has one Type II Error and no Type I Errors.
3. Panel three has three true positives, five false negatives, and two true negatives. Thus it has five Type II Errors.

(b) How many false positives, false negatives, true positives, true negatives, Type I errors, and Type II errors result from applying the Holm procedure to control the FWER at level $\alpha = 0.05$?

1. Panel one has seven true positives, one false negative, and two true negatives. Thus it has One Type II Error and no Type I Errors.
2. Panel two has eight true positives, and two true negatives. Thus it has no Type I Errors or Type II Errors.
3. Panel two has eight true positives, and two true negatives. Thus it has no Type I Errors or Type II Errors.

(c) What is the false discovery rate associated with using the Bonferroni procedure to control the FWER at level $\alpha = 0.05$?

The False discovery rate associated with using the Bonferroni procedure to control the FWER at level $\alpha = 0.05$ is 0 for all three panels since there are no true null hypotheses being rejected.

(d) What is the false discovery rate associated with using the Holm procedure to control the FWER at level $\alpha = 0.05$?

The FDR associated with using the Holm procedure to control the FWER at level $\alpha = 0.05$ is 0 since there are no false positives.

(e) How would the answers to (a) and (c) change if we instead used the Bonferroni procedure to control the FWER at level $\alpha = 0.001$?

a) At $\alpha = 0.001$, both panel one and two would have three false negatives (Type II Errors), two true negatives, and five true positives. Panel three would have five false negatives (Type II Errors), two true negatives, and three true positives. No panel has false positives (Type I Errors).

b) The FDR would remain at 0 since there is still no false positives.

Conceptual Problem 2 (2 pts)

Suppose that we test $m = 1000$ independent null hypotheses, of which 10% are true, at significance level $\alpha = 0.05$ and achieve a false discovery rate of $q = 0.20$. Construct a table following Table 13.2 in the textbook, identifying the appropriate values of V, S, U, W, and R in this situation. Answer:

$m_0 = 100$

$V = \alpha * m_0 = 5$

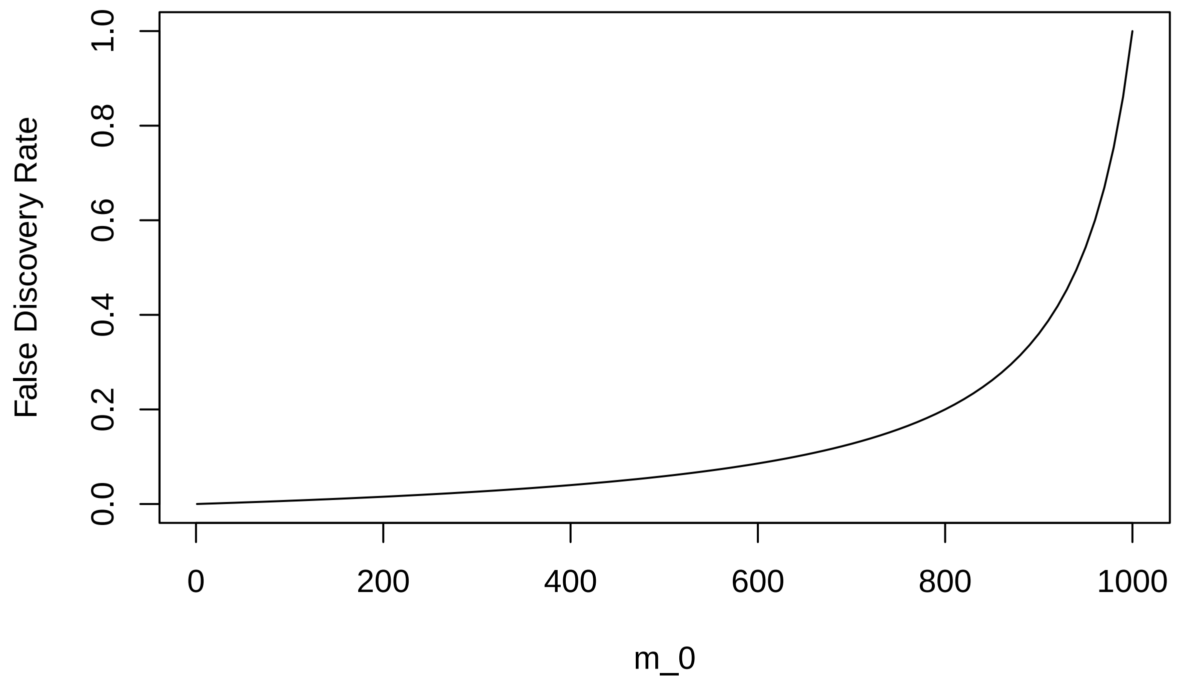
$0.2 = V/(V + S)$ -> $S = 20$

	H_0 True	H_0 False	Total
Reject H_0	5	20	25
Do not reject H_0	95	880	975
Total	100	900	1000

Conceptual Problem 3 (3 pts)

Suppose that we test $m = 1000$ independent null hypotheses, of which an unknown number m_0 are true, at significance level $\alpha = 0.05$. Suppose that each test also has a power of 0.80. Find and plot the false discovery rate as a function of m_0. Answer:

```r
al = 0.05
fdr = function(m0){al*m0/(al*m0 + 800 - 0.8*m0)}
plot(fdr, 1, 1000, xlab = "m_0", ylab = "False Discovery Rate")
```



Conceptual Problem 4 (2.5 pts)

Textbook Exercise 5.4.2 parts (a), (b), and (c).:

We will now derive the probability that a given observation is part of a bootstrap sample. Suppose that we obtain a bootstrap sample from a set of n observations. (a) What is the probability that the first bootstrap observation is not the jth observation from the original sample? Justify your answer. Answer: The probability of the first bootstrap observation not being the jth observation seems very likely. Thus logically the odds of any one observation being the jth is 1/n, so the odds of it not being the jth must then be 1 - 1/n.

(b) What is the probability that the second bootstrap observation is not the jth observation from the original sample? Answer: Using the same logic as the previous problem the probability of the second observation being the jth observation is also 1/n. Then the probability that it is not is also 1 - 1/n.

(c) Argue that the probability that the jth observation is not in the bootstrap sample is (1 - 1/n)^n.

Answer: Since we are using a bootstrap sample the samples are then independent. Meaning that the probability of jth observation not being in the bootstrap sample is the same as the previous two questions but all n times, thus the formula is (1-1/n)^n

Simulation Problems

Simulation Problem 1 (Code: 1 pt; Explanation: 0.5 pts)

Textbook Exercise 5.4.2 parts (e), (g), and (h). For part (g), you should create a line plot (using either `plot` with argument `type = "l"` or `geom_line`). Then, to make clearer what you should be commenting on, find the limit as $n \to \infty$ of the probability that the j^{th} observation is in your bootstrap sample and add a horizontal red line (using `abline` or `geom_hline`) at that value. (Hint: the limit as $n \to \infty$ of the expression in part (c) is well-known and easily found on the Internet.)

e) Using the formula in the previous question and plugging in 100, we can see our probability of the jth observation being in our bootstrap sample is 1-(1-(1/100))^100 = 0.634

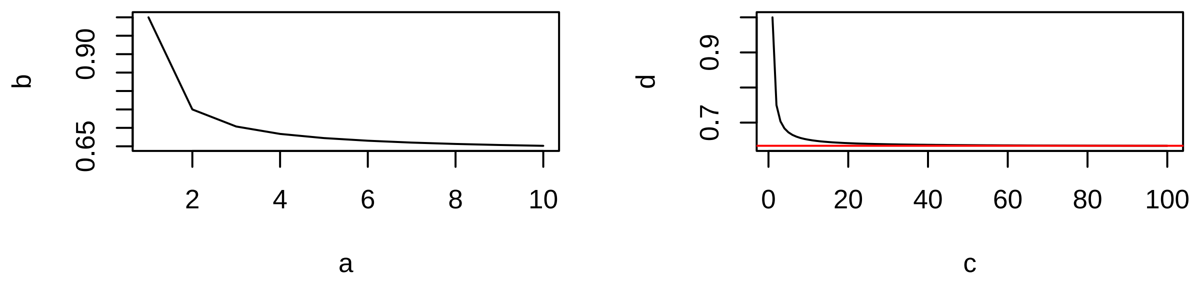
f)

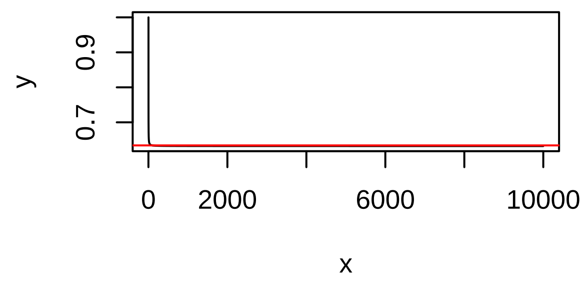
```
a = 1:10
b <- 1-(1-(1/a))^a
c = 1:100
d <- 1-(1-(1/c))^c
x = 1:10000
y <- 1-(1-(1/x))^x


par(mfrow=c(2,2))
plot(a,b, type = "l")
abline(h=.634, col = "red")

plot(c,d, type = "l")
abline(h=.634, col = "red")

plot(x,y, type = "l")
abline(h=.634, col = "red")
```





Here you can see that our probability quickly converges, especially when n is large, to .634

h)

```
store <- rep(NA, 10000)
for (i in 1:10000){
  store[i] <- sum(sample(1:100, rep = TRUE) ==4)>0
}
mean(store)
```

```
## [1] 0.626
```

This result makes a lot of sense as the probability of of it being the 4th observation is the same as it being any jth observation, we are just reaffirming that idea through simulation. This result should and does fortunately follow the same result as part e!

Simulation Problem 2 (Code: 1.5 pts; Explanation: 3.5 pts)

Copy the *functions* you created in the Bootstrap Confidence Intervals class activity as well as Simulation Parts 3, 4, and 5.

```
bootstrap_resample <- function(data_vector, B, summary_fn = mean,
                               seed = 100, ...){
  # data_vector: a vector of data
  # B: the number of bootstrap resamples
  # summary_fn: the name of a function to apply to the resampled data
  # ...: any additional arguments to summary_fn

  boot_samples <- matrix(0, nrow = length(data_vector), ncol = B)
```

```r
  n <-length(data_vector)
  # We need to add some code here!
  set.seed(seed)
  for (i in 1:B) {
     boot_samples[,i] <- sample(data_vector, size = n, replace = TRUE)

  }

  boot_stat <- apply(boot_samples, 2, summary_fn, ...)
  # Seriously, do not name any arguments x or FUN when using apply within a function
  # Advice from someone who spent 30 minutes debugging a sample solution

  return(boot_stat)
}
```

```r
bootstrap_ci <- function(data_vector, method, B = 1000, seed = 100, C = 0.95, summary_fn = mean, ...){
  # data_vector: a vector of data
  # method: the CI method
  # B: the number of bootstrap resamples
  # seed: the seed to use
  # C: the confidence level as a decimal
  # summary_fn: the name of a function to apply to the resampled data
  # ...: any additional arguments to summary_fn



  obs_stat <- do.call(summary_fn, args = list(data_vector, ...))
  # do.call allows you to call a function without having to hard-code what that function is
  # the args argument is a list of arguments to the function
  # so this will find the observed value of the statistic given the original data vector

  bootstrap_values <- bootstrap_resample(data_vector, B, summary_fn, ...)

  alpha <- 1 - C

  if (method == "percentile"){
      # write code to get the percentile confidence level out of the returned bootstrap_values and stor

    boot_ci <- quantile(bootstrap_values,
                        probs = c(alpha/2, 1 - alpha/2)
                        )
  } else if (method == "basic"){
    # write code to get the "basic" confidence level and store in a length 2 vector boot_ci
    boot_ci <- 2 * obs_stat -
      quantile(bootstrap_values, probs = c(1 - alpha/2, alpha/2))
  } else if (method == "normal"){
    # write code to get the normal-theory confidence interval and store in a length 2 vector boot_ci
    # make sure to use the adjustments in the course notes, e.g., don't use 1/sqrt(n) as your standard
    center <- 2*obs_stat - mean(bootstrap_values)
    crit_value <- qnorm(alpha/2)
    se_boot <- sd(bootstrap_values)
    boot_ci <- center + c(1, -1) * crit_value * se_boot
  }

  return(boot_ci)
}
```

Simulation Part 3 - Assumptions Not Met, Large Samples

Simulate 1000 sets of 100 random numbers from a $Exp(1)$ distribution and store them in a 1000 x 100 matrix `sim_data3`. Note that `sim_data3` represents 1000 samples of size 100 under conditions where we *know* the assumptions of a t confidence interval are not met but we are hoping that the Central Limit Theorem can cover us.

```
set.seed(437)
# need to do the simulation now!
sim_data3 <- matrix(rexp(1000*100), nrow = 1000, ncol = 10)
```

Copy and modify your code chunks from Simulation Part 1 to produce 95% confidence intervals using `sim_data3`. What proportion of "95% confidence intervals" actually contained the true parameter value? Did any methods perform noticeably better/worse?

```
# You should be able to just run this code chunk without any fixes
pop_mean3 <- 1
ci_t3 <- apply(sim_data3, 1, function(x) t.test(x)$conf.int)
ci_t_df3 <- as.data.frame(t(ci_t3))
names(ci_t_df3) <- c("lower", "upper")
ci_t_df3 %>% mutate(covered = lower <= pop_mean3 & upper >= pop_mean3) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.903
```

```
# You should be able to just run this code chunk without any fixes
ci_perc3 <- apply(sim_data3, 1, bootstrap_ci, method = "percentile", B = 1000, summary_fn = mean, na.rm
# don't need any ... arguments here, but illustrating the idea of the ...
# notice that bootstrap_ci has default seed = 100 and C = 0.95 arguments
ci_perc_df3 <- as.data.frame(t(ci_perc3))
names(ci_perc_df3) <- c("lower", "upper")
ci_perc_df3 %>% mutate(covered = lower <= pop_mean3 & upper >= pop_mean3) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.857
```

Copy and modify the chunk above for the basic and normal-theory intervals.

```
ci_basic3 <- apply(sim_data3, 1, bootstrap_ci, method = "basic", B = 1000, summary_fn = mean, na.rm = T

ci_basic_df3 <- as.data.frame(t(ci_basic3))
names(ci_basic_df3) <- c("lower", "upper")
ci_basic_df3 %>% mutate(covered = lower <= pop_mean3 & upper >= pop_mean3) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.845
```

```
ci_normal3 <- apply(sim_data3, 1, bootstrap_ci, method = "normal", B = 1000, summary_fn = mean, na.rm =
ci_normal_df3 <- as.data.frame(t(ci_normal3))
names(ci_normal_df3) <- c("lower", "upper")
ci_normal_df3 %>% mutate(covered = lower <= pop_mean3 & upper >= pop_mean3) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.856
```

Simulation Part 4 - Assumptions Not Met, Small Samples

Simulate 1000 sets of 10 random numbers from a $Exp(1)$ distribution and store them in a 1000 x 100 matrix `sim_data4`. Note that `sim_data4` represents 1000 samples of size 10 under conditions where we *know* the assumptions of a t confidence interval are not met and Central Limit Theorem is almost certainly *not* going to cover us.

```
set.seed(437)
# need to do the simulation now!
sim_data4 <- matrix(rexp(1000*10), nrow = 1000, ncol = 10)
```

Copy and modify your code chunks from Simulation Part 1 to produce 95% confidence intervals using `sim_data4`. What proportion of "95% confidence intervals" actually contained the true parameter value? Did any methods perform noticeably better/worse?

All of the methods performed worse as breaking the central limit theorem was still an issue even though bootstrapping tends to ignore it. The range of values was around 84-86% with the t test being noticeably better at 90%.

```
pop_mean4 <- 1
ci_t4 <- apply(sim_data4, 1, function(x) t.test(x)$conf.int)
ci_t_df4 <- as.data.frame(t(ci_t4))
names(ci_t_df4) <- c("lower", "upper")
ci_t_df4 %>% mutate(covered = lower <= pop_mean4 & upper >= pop_mean4) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.903
```

```
ci_perc4 <- apply(sim_data4, 1, bootstrap_ci, method = "percentile", B = 1000, summary_fn = mean, na.rm
ci_perc_df4 <- as.data.frame(t(ci_perc4))
names(ci_perc_df4) <- c("lower", "upper")
ci_perc_df4 %>% mutate(covered = lower <= pop_mean4 & upper >= pop_mean4) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.857
```

```
ci_basic4 <- apply(sim_data4, 1, bootstrap_ci, method = "basic", B = 1000, summary_fn = mean, na.rm = T

ci_basic_df4 <- as.data.frame(t(ci_basic4))
names(ci_basic_df4) <- c("lower", "upper")
ci_basic_df4 %>% mutate(covered = lower <= pop_mean4 & upper >= pop_mean4) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.845
```

```
ci_normal4 <- apply(sim_data4, 1, bootstrap_ci, method = "normal", B = 1000, summary_fn = mean, na.rm =
```

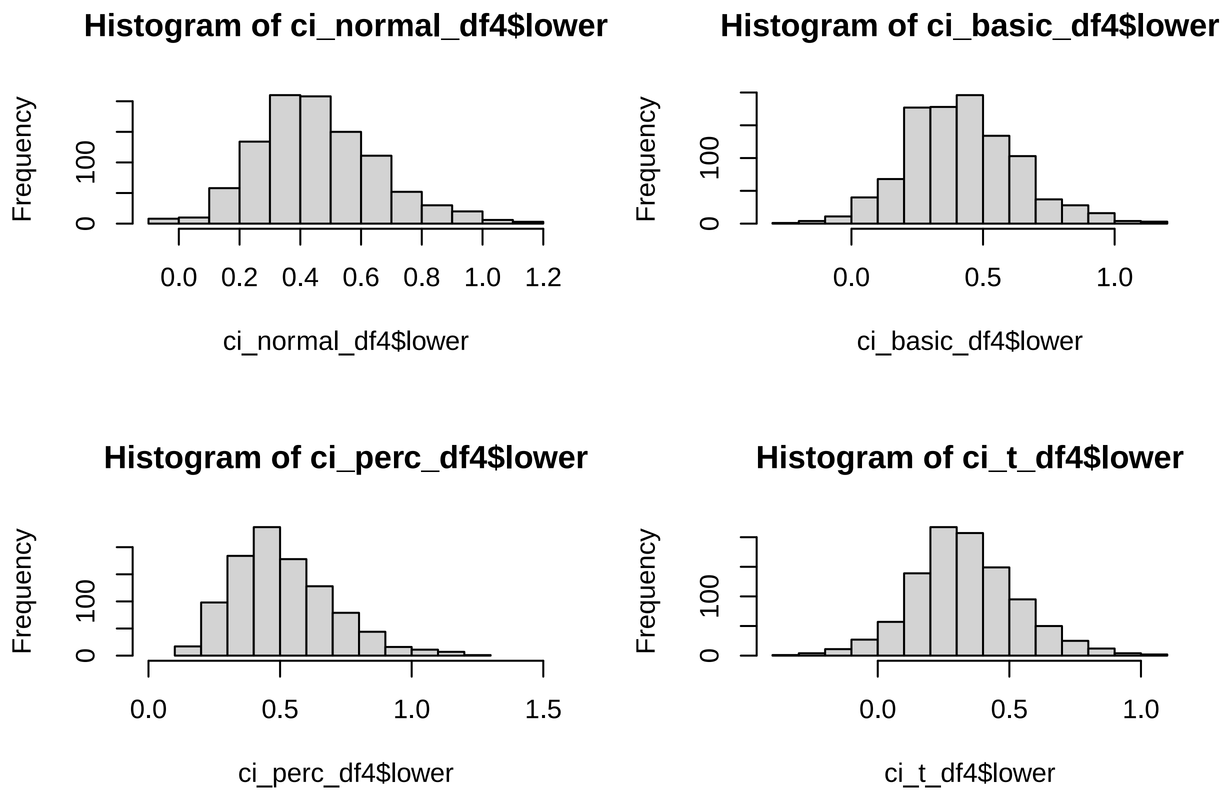
```r
ci_normal_df4 <- as.data.frame(t(ci_normal4))
names(ci_normal_df4) <- c("lower", "upper")
ci_normal_df4 %>% mutate(covered = lower <= pop_mean4 & upper >= pop_mean4) %>%
  summarize(coverage_probability = mean(covered))
```

```
##   coverage_probability
## 1                0.856
```

Simulation Part 5 - Range Preservation

Produce a histogram of the lower bound of the CIs you obtained using each of the four methods in Simulation Part 4. Which methods appear to be range-preserving (that is, cannot give implausible values for the parameter) even under this "worst-case scenario"?

```r
par(mfrow= c(2,2))
hist(ci_normal_df4$lower)
hist(ci_basic_df4$lower)
hist(ci_perc_df4$lower, xlim = c(0, 1.5))
hist(ci_t_df4$lower)
```



Write a brief summary of what you learned from the activity. Make sure to address the following questions:

- Are theory-based methods *guaranteed* to achieve the appropriate coverage? What about bootstrap-based methods?

Theory-based methods are not guaranteed to achieve the appropriate coverage but they can achieve appropriate coverage for populations that are normally distributed or meet the requirements for the Central Limit Theorem.

With bootstrap-based methods we are not guaranteed appropriate coverage as was seen in the simulations where we consistently had coverage probabilities around 10% lower than the appropriate coverage.

- Which of the four methods appear to be range-preserving even in a “worst-case scenario”?

The percentile method appears to be range-preserving even in a “worst-case scenario” since it does not give implausible values (< 0) for the parameter.

- When and why would a bootstrap method be useful to obtain a confidence interval even if it doesn’t achieve the appropriate coverage?

Bootstrap method would be useful to obtain a confidence interval when the distribution assumed by the theory-based methods do not accurately reflect the distribution of our statistic or when we are unsure about the distribution, since bootstrapping allows us to estimate the distribution based on our sample.

Applied Problems

Applied Problem 1 (Code: 4 pts; Explanation: 2 pts)

Using the `dplyr` package, subset the `mpg` dataset from the `ggplot2` package to include only the cars from 2008 that are minivans, pickups, or SUVs (`%in%` is a useful replacement for `==` when trying to match to more than one possibility). Using this new dataset, determine which of the following statements is/are true, using an $\alpha = 0.10$ significance level/family-wise error rate or a $q = 0.10$ false discovery rate:

```
filtered_data <- mpg %>%
  filter(year == 2008,
         class %in% c("minivan", "pickup", "suv"))
```

1. There is a significant difference in highway gas mileage between minivans and SUVs. H0: There is no significant difference between gas mileage between minivans and SUVs Ha: There is a significant difference between gas mileage between minivans and SUVs
2. There is a significant difference in highway gas mileage between pickups and SUVs. H0: There is not a significant differencce in highway gas mileage between pickups and SUVs Ha: There is a significant differencce in highway gas mileage between pickups and SUVs
3. There is a significant difference in highway gas mileage between minivans and pickups. H0: There is a not significant difference in highway gas mileage between minivans and pickups. Ha: There is a significant difference in highway gas mileage between minivans and pickups. Use the following methods.

(a) Three two-sample t-tests with no adjustments for multiple testing. Store all three p-values in a single vector so that you can use the `p.adjust` function in later parts.

```
part_1_data <- mpg %>%
  filter(year == 2008,
         class %in% c("minivan", "suv"))

part_2_data <- mpg %>%
  filter(year == 2008,
         class %in% c("pickup", "suv"))

part_3_data <- mpg %>%
  filter(year == 2008,
         class %in% c("minivan", "pickup"))
```

```
rejections <- c(t.test(hwy ~ class,
                       data = part_1_data)$p.value,
                t.test(hwy ~ class,
                       data = part_2_data)$p.value,
                t.test(hwy ~ class,
                       data = part_3_data)$p.value)
```

1. There is a significant difference in highway gas mileage between minivans and SUVs. H0: There is no significant difference between gas mileage between minivans and SUVs Ha: There is a significant difference between gas mileage between minivans and SUVs
   H0 REJECTED
2. There is a significant difference in highway gas mileage between pickups and SUVs. H0: There is not a significant differencce in highway gas mileage between pickups and SUVs Ha: There is a significant differencce in highway gas mileage between pickups and SUVs H0 REJECTED
3. There is a significant difference in highway gas mileage between minivans and pickups. H0: There is a not significant difference in highway gas mileage between minivans and pickups. Ha: There is a significant difference in highway gas mileage between minivans and pickups. H0 REJECTED

Main: 1) H0 REJECTED 2) H0 REJECTED 3) H0 REJECTED

Bonferonni: 1) H0 NOT REJECTED 2) H0 NOT REJECTED 3) H0 STILL REJECTED

Holm's 1) H0 NOT REJECTED 2) H0 NOT REJECTED 3) H0 STILL REJECTED

Tukey: 1) H0 REJECTED 2) H0 NOT REJECTED 3) H0 STILL REJECTED

Benjamini-Hochberg 1) H0 REJECTED 2) H0 REJECTED 3) H0 STILL REJECTED

Bonferonni: 1) H0 NOT REJECTED 2) H0 NOT REJECTED 3) H0 STILL REJECTED

Holm's 1) H0 NOT REJECTED 2) H0 NOT REJECTED 3) H0 STILL REJECTED

Tukey: 1) H0 REJECTED 2) H0 NOT REJECTED 3) H0 STILL REJECTED

Benjamini-Hochberg 1) H0 REJECTED 2) H0 REJECTED 3) H0 STILL REJECTED

Applied Problem 2 (Code: 1 pt; Explanation: 1 pt)

Use a one-way ANOVA followed by Scheffe's method (`ScheffeTest` in the DescTools package) to determine whether the following statement is true at the $\alpha = 0.10$ significance level:

There is a significant difference in highway gas mileage between pickups and non-pickups (SUVs and minivans).

```
lm1 <- aov(hwy ~ class, data = filtered_data)

anova(lm1)
```

```
## Analysis of Variance Table
##
## Response: hwy
##           Df Sum Sq Mean Sq F value   Pr(>F)
## class      2 110.15  55.075  5.4929 0.006851 **
## Residuals 52 521.38  10.026
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
my_anova <- aov(hwy ~ class, data = filtered_data)
summary(my_anova) # check for significant p-value
```

```
##             Df Sum Sq Mean Sq F value  Pr(>F)
## class        2  110.1   55.07   5.493 0.00685 **
## Residuals   52  521.4   10.03
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
contrast.matrix <- matrix(
c(1, -1/2, -1/2), # tests that mu1 is different from the average of mu2 and mu3
nrow = 3, ncol = 1
)
ScheffeTest(my_anova, conf.level = 0.95) # 95% confidence intervals
```

```
##
##   Posthoc multiple comparisons of means: Scheffe Test
##     95% family-wise confidence level
##
## $class
##                      diff     lwr.ci     upr.ci   pval
## pickup-minivan -5.258824 -9.3183266 -1.1993205 0.0079 **
## suv-minivan    -3.563636 -7.3929489  0.2656762 0.0732 .
## suv-pickup      1.695187 -0.6869928  4.0773671 0.2101
##
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

There is a significant difference at an alpha level of .1

Applied Problem 3 (Code: 5 pts; Explanation: 3 pts)

Textbook Exercise 5.4.9. a)

```
(mu_hat <- mean(Boston$medv))
```

```
## [1] 22.53281
```

The mean of medv in the Boston housing data set is 22.53281.

b) the standard error is equal to standard deviation / sqrt(n)

```
(sd(Boston$medv))/sqrt(nrow(Boston))
```

```
## [1] 0.4088611
```

The standard error is 0.4088611.

c)

```
mu.fn <- function(data, index){
  X <- data$medv[index]
  mean(X)
}
boot(Boston, statistic = mu.fn, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston, statistic = mu.fn, R = 1000)
##
##
## Bootstrap Statistics :
##     original       bias    std. error
## t1* 22.53281 -0.005806719   0.4219161
```

```
boot_mu_hat <- bootstrap_resample(Boston$medv, B = 1000, seed = 549)
sd(boot_mu_hat)
```

```
## [1] 0.4119417
```

The standard error of $\hat{\mu}$ using bootstrap is very similar to the standard error of the original sample mean, the bootstrapped standard error is just slightly higher.

d)

```
lower <- mu_hat - 2*sd(boot_mu_hat)
upper <-  mu_hat + 2*sd(boot_mu_hat)
(mutest <- t.test(Boston$medv)$conf.int)
```

```
## [1] 21.72953 23.33608
## attr(,"conf.level")
## [1] 0.95
```

A 95% confidence interval of $\hat{\mu}$ using bootstrap estimates is about [21.71, 23.36], which is very close in value to the 95% confidence interval from the t.test function, which is slightly narrower.

e)

```
median(Boston$medv)
```

```
## [1] 21.2
```

The median value of medv in the population is 21.2.

f)

```
boot_med_hat <- bootstrap_resample(Boston$medv, B = 1000, seed = 549, summary_fn = median)
sd(boot_med_hat)
```

```
## [1] 0.3844301
```

The standard error of $\hat{\mu}_{med}$ is 0.3844301 which is lower than both the original and bootstrapped mean of medv. This implies that it is more likely that the median does a slightly better job of representing the distribution of the population of medv.

g)

```
(mu_0.1 <- quantile(Boston$medv, probs = .10))
```

```
##   10%
## 12.75
```

h)

```
set.seed(549)
mu01.fn <- function(data, index){
  X <- data$medv[index]
  quantile(X, probs = .1)
}
boot(Boston, statistic = mu01.fn, R = 1000)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Boston, statistic = mu01.fn, R = 1000)
```

```
##
##
## Bootstrap Statistics :
##     original  bias    std. error
## t1*    12.75  0.0204    0.487608
```

The bootstrapped estimate of standard error of $\hat{\mu}_{0.1}$ is higher than all previous estimates of standard error for other parameters, implying that it is a less accurate representation of $\hat{\mu}_{0.1}$ than previous bootstrapped estimates have been of their respective $\hat{\mu}$s.