Jared Yu

Computational Statistics

Problem Set 9

*NOTE: The code for the assignment is in the Appendix.*

1. Problem 9.5 with these additional comments for each part:
   1. Only find the bootstrap confidence interval and use the bootstrap variance estimate presented in Lecture 9B and in Gentle 13.2. Don’t forget to work with the data on the log scale.

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

* 1. This will be an application of the randomization method presented in Module 8 and discussed in Gentle 12.1. It is referred to as a permutation test in the text and is discussed in Section 9.8 of the Givens and Hoeting Text.

Let the log stomach data from the table be represented by where and let the log breast data from the table be presented by where .

Using R, the statistic , where , is calculated to be . After doing the permutation test where combinations of the data from and were created and the corresponding for were calculated. The value of is ranked as the value. Therefore, we reject at the significance level. The conclusion then is that at significance level .

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. Use the percentile method to find the required confidence intervals. You will need to exponentiate the intervals from (a) in order to compare them.

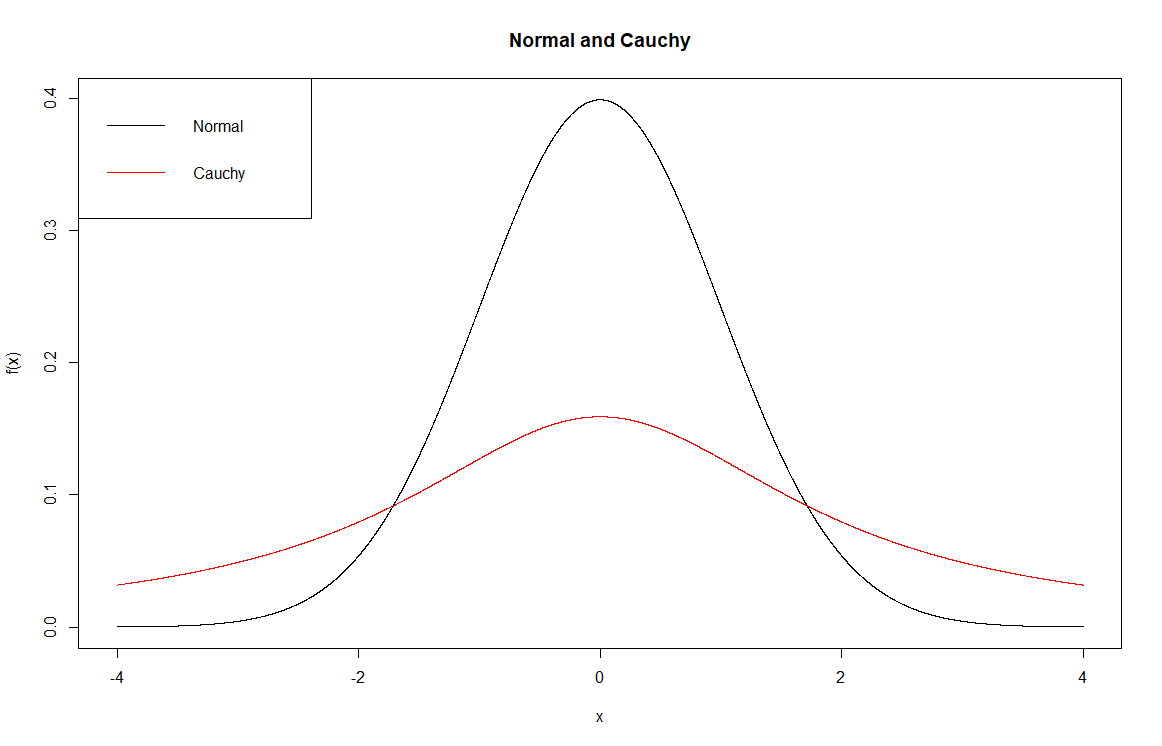
Using the percentile method and exponentiating the boundaries (exponentiated percentile C.I. (log data)), the resulting bounds of for the breast data and for the stomach data are obtained. In the breast data, the boundary fits around the original mean of , however, in the stomach data the bounds are to the left of the original mean of . When exponentiating the bounds of the bootstrap confidence intervals (exponentiated bootstrap t C.I.), they both (breast and stomach data) have the same problem of being to the left of the original mean. However, when the percentile method is used on the original data (percentile C.I. (original data)), there’s no issue with either dataset having the confidence intervals fit around the mean of the original dataset.

It seems then that there’s an issue with taking the log of the data, finding a confidence interval, and then exponentiating the bounds to look back at the original values. Then a possible issue is that it’s important to stay within the log scale and to perform re-calculations if changing from log scale to original scale.

1. Problem 9.7.

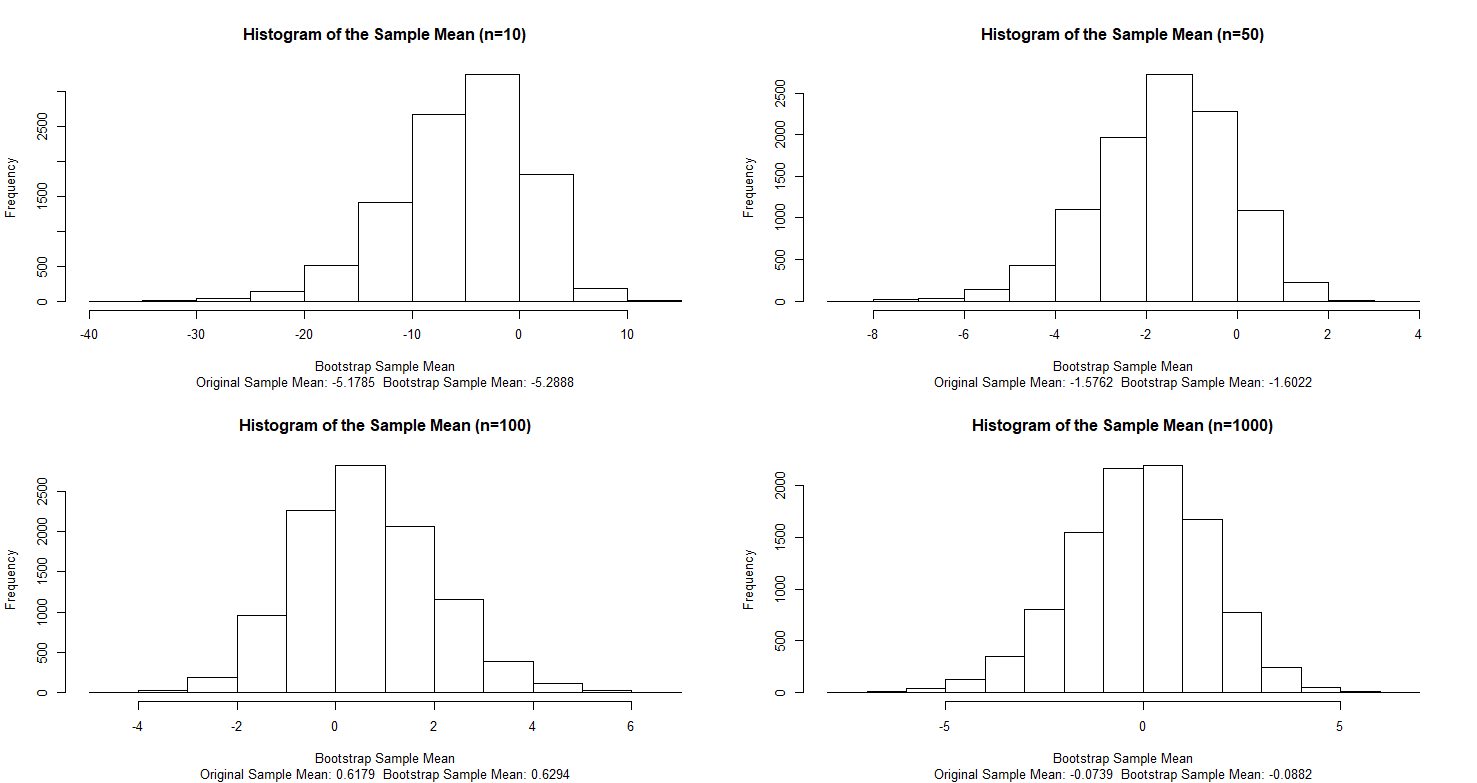
**Cauchy example**

The Cauchy distribution is a heavy-tailed distribution. Below in Figure 1 is a plot of the standard Normal distribution in black and a Cauchy distribution in red. The parameters for the Cauchy are (location) and (scale). Compared to the standard normal with and , the tails are visibly much heavier.



Figure

The problem with the principle of bootstrapping is that the given sample is ideally reflective of the population. Therefore, if a sample derived from the Cauchy distribution isn’t large enough, then it may lead to poor bootstrapping results due to the (increased) possibility of observations from the heavy tails over-representing the data in the sample.

The following examples will sample and points from a Cauchy distribution with the same parameters as those shown in the graph (the seed is 999). The bootstrap process used for the total number of bootstrap samples. In Figure 2 on the top-left is the histogram of the bootstrap sample means for when . In this case, it’s evident that the sample means are skewed towards the left and are further away from where the above graph in Figure 1 indicated they instinctively would be (around 0). However, due to the heavy tails, the sample size of isn’t large enough to generate a representative sample that can be thought of as similar to the true population. The sample mean of this Cauchy sample is approximately . Such a value shows already that the sample itself is quite skewed and so bootstrapping from this single case is a poor choice. The mean from bootstrapping is , which is also indicative of the sample size problem.

Figure

In the top-right plot of Figure 2, it shows the case where bootstrapping is done for observations in a sample. Again, looking at the sample mean of approximately shows that there’s an issue with the sample itself being properly representative of the entire population. Looking at the histogram, there’s a skew towards the left. The bootstrap mean is and it also shows the same problems as the sample mean from the original sample size of .

At the bottom-left of Figure 2 is the case where bootstrapping was done for a sample size of . The sample mean this time is closer to , at approximately . Looking at the histogram however, it still shows that there’s a skew, but this time it’s to the right. The bootstrap mean is and it’s evident in the histogram that there’s an apparent skew in the bootstrap data.

Finally, at the bottom-right of Figure 2 is the case where bootstrapping was done with observations. When taking a sample of this size for the Cauchy distribution, the original sample mean of approximately seems to indicate that the sample has reached a size large enough where it can begin to become representative of the entire population. There’s no longer the case of observations from the tails dominating the distribution of the sample. The histogram itself looks far more symmetric and is centered around . The bootstrap mean of shows too that there’s finally a sample size large enough to be representative of the true population.

**Uniform example**

Another situation is that of parameter estimation using bootstrap with the Uniform distribution. In this case, the goal is to estimate from a distribution. First, the goal will be to show how can be estimated from a single sample.

Let , then ; . The support in this case contains so the normal method of obtaining the MLE will not work.

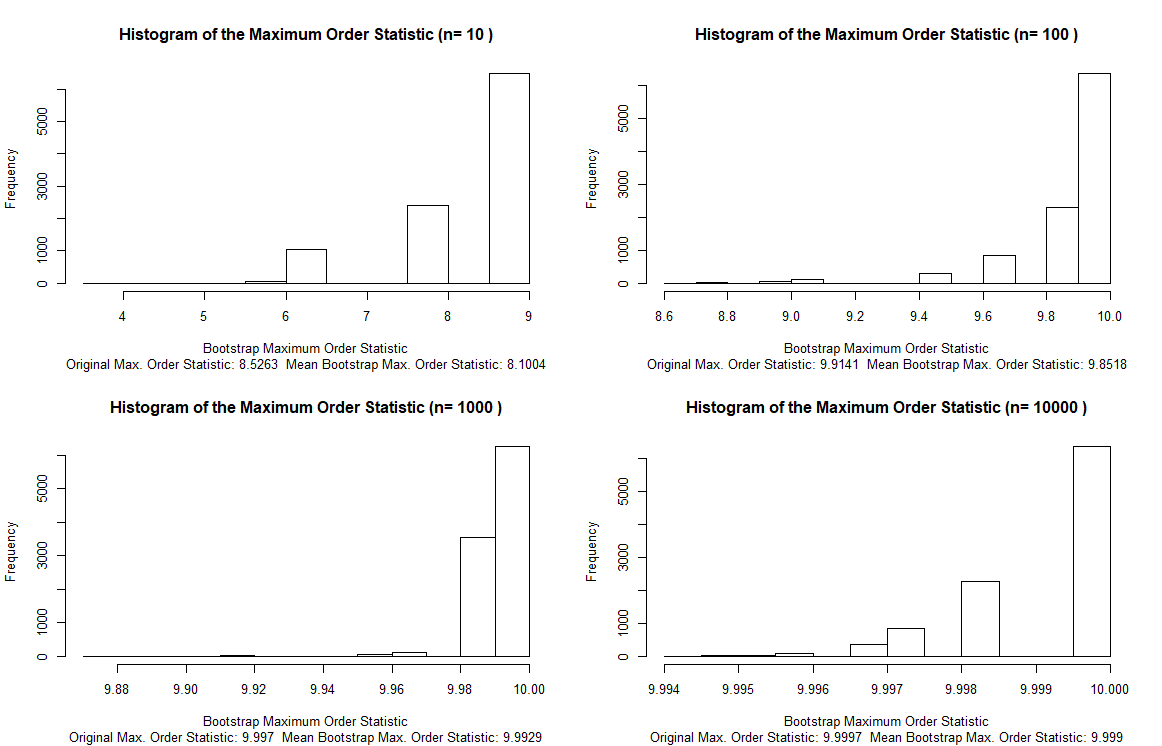
where

When , (where represents the maximum or order statistic). Therefore, the only need is to look at cases with . The j.p.d.f. can therefore be rewritten as follows:

When decreases, increases. So, to maximize the minimum value for must be chosen where . .

Next, the problem of the necessity for extreme values (in this case, the maximum order statistic) from the in order to obtain the MLE will be highlighted. Below in Figure 3 is a grid of histograms of the bootstrap MLE, for (), for different original sample sizes of and . (*NOTE:* The original maximum order statistic was calculated from an original sample of , however, the value of in this case is arbitrary and is only used to show the contradiction in trying to bootstrap when there’s an emphasis on extreme values.)

The issue is that in order to calculate , it’s necessary to calculate from the bootstrap sample. However, with the bootstrap principle the largest value from the original sample will determine the largest possible value for all bootstrap samples. Therefore, the resulting from each of the bootstrap samples can never be larger than the original . The resulting appearance of each of the histograms of for is an exponential distribution with an extreme skew to the left and a spike on the right-hand side near the of the original sample. If, however, the value of were calculated with the mean or the median, this type of situation wouldn’t occur since they’re far less dependent on extreme values.



Figure

1. Let be a random sample from a population with mean , variance , and distribution . Let be the empirical distribution function. Let be the sample mean for . Let be a random sample taken with replacement from . Let be the sample mean for .
   1. Show that

First define a functional of the population distribution, , along with a parameter where

In this case, , , and . Therefore, the above can be rewritten where the functional of the population is and the parameter can be defined as,

Likewise, it follows that for the empirical distribution function, that,

Then the following holds true:

* 1. Show that

Using the same rules that were mentioned in part (a), the following also holds true:

1. In this problem we will compare the results of a normal bootstrap sample to those of a balanced bootstrap sample.
   1. Draw a random sample of size from the standard Normal distribution, , and calculate the sample mean, .

First needs to be calculated.

Using a seed of 999 in R, the sample mean is calculated to be

* 1. Use the standard bootstrap method to generate bootstrap pseudodata sets. For each data set calculate the sample mean, . Then find the bootstrapped bias corrected estimator, , and the bootstrapped estimation of the variance, .

where

The result using R is

While .

The bootstrapped estimation of the variance, is calculated as follows:

The result using R is

* 1. Repeat part (b) using the balanced bootstrap method.

In this case, the following occurred:

(with the exception that this time, values are obtained through the balanced bootstrap method)

While .

Likewise, using values obtained through the balanced bootstrap method, is calculated as follows:

* 1. Discuss your results.

Utilizing the balanced bootstrap method, the bias of the estimate is reduced to , while using normal bootstrap the bias of the same estimate is approximately . This result is expected, since when using balanced bootstrap, the values in the bootstrap samples, , occur with the same relative frequency as they do in the original sample . Therefore, the (the average of the balanced bootstrap sample means) will have the same value as (the original sample mean). This eliminates a source of potential Monte Carlo error.

In reducing the bias of the estimator, the variance increases from approximately to . The increase in variance is quite small compared to the reduction in bias. Therefore, it seems that in this case it’s worth using the balanced bootstrap method to reduce the bias of the estimator .

Another interesting note is that the bias of the parameter () itself also increases from approximately to . Also, the notes mention that should be less than which in both cases it is not. However, in the case of the balanced bootstrap it shouldn’t be expected that would be smaller since is .

Appendix

### Problem 1  
# part (a)  
stomach <- c(25, 42, 45, 46, 51, # Load data  
 103, 124, 146, 340, 396, 412, 876, 1112)  
breast <- c(24, 40, 719, 727, 791,  
 1166, 1235, 1581, 1804, 3460, 3808)  
log\_stomach <- log(stomach)  
log\_breast <- log(breast)  
  
par(mfrow=c(1,2))  
boxplot(stomach)  
boxplot(breast)  
dev.off()  
  
par(mfrow=c(1,2))  
boxplot(log(stomach))  
boxplot(log(breast))  
dev.off()  
  
mean(log\_stomach) # 4.96792  
mean(log\_breast) # 6.558603  
  
### Finds the bootstrap t C.I. on log scale  
bootstrap\_t <- function(df, B = 1e3, alpha = 0.05) {  
 log\_df <- log(df) # Take log scale  
 n <- length(df) # Initialize variables  
 T\_F <- mean(log\_df)  
 data\_matrix <- matrix(NA, nrow = B, ncol = length(log\_df))  
 theta\_matrix <- matrix(NA, nrow = B)  
 R\_X\_F\_matrix <- matrix(NA, nrow = B)  
 set.seed(999) # Set seed  
 for (i in 1:B) { # Bootstrap sample  
 data\_matrix[i,] <- sample(log\_df, size = length(log\_df), replace = TRUE)  
 theta\_matrix[i] <- mean(data\_matrix[i,])  
 V\_F\_star <- (1 / (n - 1)) \* sum((data\_matrix[i,] - theta\_matrix[i])^2)  
 R\_X\_F\_matrix[i] <- (theta\_matrix[i] - T\_F) / sqrt(V\_F\_star)  
 }  
   
 theta\_bar <- mean(theta\_matrix) # mean bootstrap theta  
 # variance of bootstrap theta  
 var\_theta\_hat\_star <- sum((theta\_matrix - theta\_bar)^2) / (B - 1)  
   
 # Find quantiles  
 xi <- quantile(R\_X\_F\_matrix, probs = c(alpha / 2, 1 - alpha / 2))  
 names(xi) <- NULL  
 bootstrap\_t <- c(T\_F - sqrt(var\_theta\_hat\_star) \* xi[2],  
 T\_F - sqrt(var\_theta\_hat\_star) \* xi[1])  
 return(bootstrap\_t)  
}  
stomach\_bootstrap\_t\_CI <- bootstrap\_t(df = stomach, B = 1e4, alpha = 0.05) # 4.772891 5.195032  
breast\_bootstrap\_t\_CI <- bootstrap\_t(df = breast, B = 1e4, alpha = 0.05) # 5.891237 6.803640  
  
# part (b)  
# H\_0: no diff. in mean survival times (mu\_log\_breast = mu\_log\_stomach)  
# H\_1: there is a diff. in mean survival times (mu\_log\_breast =/= mu\_log\_stomach)  
n1 <- length(log\_stomach)  
n2 <- length(log\_breast)  
  
t0 <- mean(log\_stomach) - mean(log\_breast)  
m <- choose((n1 + n2), n2)  
m\_combinations <- t(combn(c(log\_stomach, log\_breast), n2)) # m combinations of both data  
stomach\_breast <- c(log\_stomach, log\_breast) # Combined data  
t\_matrix <- matrix(NA, nrow = m) # Empty matrix for t\_i  
  
for (i in 1:m) { # Calculate t\_i for all m combinations of the data  
 ys <- m\_combinations[i,]  
 # Reference: https://stackoverflow.com/questions/5812478/how-i-can-select-rows-from-a-dataframe-that-do-not-match  
 xs <- subset(stomach\_breast, !(stomach\_breast %in% ys))  
 xbar\_new <- mean(xs); ybar\_new <- mean(ys)  
 t\_matrix[i] <- xbar\_new - ybar\_new  
}  
boxplot(t\_matrix)  
k <- which(t0 == sort(t\_matrix, decreasing = TRUE))  
k / m # 0.9925185  
# rej. H\_0 with sig. level. 99.25%  
  
# part (c)  
# percentile method  
percentile\_method <- function(df = breast, B = 1e3, alpha = 0.05, take\_log = TRUE) {  
 if (take\_log == TRUE) {  
 log\_df <- log(df) # Take log scale  
 } else {  
 log\_df <- df  
 }  
 data\_matrix <- matrix(NA, nrow = B, ncol = length(log\_df))  
 theta\_matrix <- matrix(NA, nrow = B)  
 set.seed(888) # Set seed  
 for (i in 1:B) { # Bootstrap sample  
 data\_matrix[i,] <- sample(log\_df, size = length(log\_df), replace = TRUE)  
 theta\_matrix[i] <- mean(data\_matrix[i,])  
 }  
   
 percentile\_CI <- quantile(theta\_matrix, probs = c(alpha / 2, 1 - alpha / 2))  
 names(percentile\_CI) <- NULL  
   
 return(percentile\_CI)  
}  
breast\_percentile\_CI <- percentile\_method(df = breast) # 5.617261 7.375876  
exp(breast\_percentile\_CI) # 275.1347 1596.9896  
exp(breast\_bootstrap\_t\_CI) # 361.8524 901.1214  
mean(breast) # 1395.909  
mean(log\_breast) # 6.558603  
breast\_percentile\_CI\_nonlog <- percentile\_method(df = breast, # 778.1273 2145.3841  
 take\_log = FALSE)  
  
stomach\_percentile\_CI <- percentile\_method(df = stomach) # 4.352572 5.631810  
exp(stomach\_percentile\_CI) # 77.67798 279.16689  
exp(stomach\_bootstrap\_t\_CI) # 118.2606 180.3738  
mean(stomach) # 286  
mean(log\_stomach) # 4.96792  
stomach\_percentile\_CI\_nonlog <- percentile\_method(df = stomach, # 133.30 492.25  
 take\_log = FALSE)  
### Problem 2  
# Cauchy distribution  
xs <- seq(-4, 4, length.out = 1e4)  
plot(xs, dnorm(x = xs, mean = 0, sd = 1), type = 'l',  
 main = 'Normal and Cauchy', xlab = 'x', ylab = 'f(x)')  
lines(xs, dcauchy(x = xs, location = 0, scale = 2), type = 'l', col = 'red')  
legend("topleft", legend = c('Normal', 'Cauchy'), col = c('black', 'red'), lty = c(1,1))  
  
par(mfrow=c(2,2))  
cauchy\_bootstrap <- function(sample\_size = 10, B = 1e4, alpha = 0, beta = 2) {  
 set.seed(888)  
 cauchy\_sample <- rcauchy(n = sample\_size, location = alpha, scale = beta) # Original sample  
 data\_matrix <- matrix(NA, nrow = B, ncol = length(cauchy\_sample)) # Initialize variables  
 theta\_matrix <- matrix(NA, nrow = B)  
 set.seed(888) # Set seed  
 for (i in 1:B) { # Bootstrap sample  
 data\_matrix[i,] <- sample(cauchy\_sample, size = length(cauchy\_sample), replace = TRUE)  
 theta\_matrix[i] <- mean(data\_matrix[i,])  
 }  
 hist(theta\_matrix, main = paste('Histogram of the Sample Mean (n=',sample\_size,')'),  
 xlab = 'Bootstrap Sample Mean',  
 sub = paste('Original Sample Mean:', round(mean(cauchy\_sample), 4),  
 ' Bootstrap Sample Mean:', round(mean(theta\_matrix), 4)))  
}  
cauchy\_bootstrap(sample\_size = 1e1)  
cauchy\_bootstrap(sample\_size = 5e1)  
cauchy\_bootstrap(sample\_size = 1e2)  
cauchy\_bootstrap(sample\_size = 1e3)  
  
# Uniform distribution  
xs <- seq(0, 10, length.out = 1e4)  
plot(xs, dunif(xs, 0, 1e4), type = 'l', ylim = c(0, 0.00014))  
segments(x0 = c(0,10), y0 = 0, x1 = c(0,10), y1 = dunif(xs, 0, 1e4))  
abline(h = 0)  
  
theta <- 10  
set.seed(999)  
unif\_sample <- runif(n = 1e1, min = 0, max = theta)  
sample(unif\_sample, size = length(unif\_sample), replace = TRUE)  
theta\_hat <- max(unif\_sample)  
  
unif\_bootstrap <- function(sample\_size = 100, B = 1e4, theta\_max = 10) {  
 set.seed(999) # Set seed  
 unif\_sample <- runif(n = sample\_size, min = 0, max = theta\_max) # Original sample  
 data\_matrix <- matrix(NA, nrow = B, ncol = sample\_size) # Initialize variables  
 theta\_matrix <- matrix(NA, nrow = B)  
 set.seed(999) # Set seed  
 for (i in 1:B) { # Bootstrap sample  
 data\_matrix[i,] <- sample(unif\_sample, size = sample\_size, replace = TRUE)  
 theta\_matrix[i] <- max(data\_matrix[i,])  
 }  
 hist(theta\_matrix, main = paste('Histogram of the Maximum Order Statistic (n=',sample\_size,')'),  
 xlab = 'Bootstrap Maximum Order Statistic',  
 sub = paste('Original Max. Order Statistic:', round(max(unif\_sample), 4),  
 ' Mean Bootstrap Max. Order Statistic:', round(mean(theta\_matrix), 4)))  
}  
par(mfrow = c(2,2))  
unif\_bootstrap(sample\_size = 1e1, B = 1e4)  
unif\_bootstrap(sample\_size = 1e2, B = 1e4)  
unif\_bootstrap(sample\_size = 1e3, B = 1e4)  
unif\_bootstrap(sample\_size = 1e4, B = 1e4)  
dev.off()  
  
### Problem 4  
# part (a)  
set.seed(999)  
sample\_size <- 1e2  
random\_sample <- rnorm(n = sample\_size, mean = 0, sd = 1)  
sample\_mean <- mean(random\_sample)  
  
# part (b)  
B <- 1e1  
set.seed(999) # Set seed  
data\_matrix <- matrix(NA, nrow = B, ncol = sample\_size) # Initialize variables  
theta\_matrix <- matrix(NA, nrow = B)  
set.seed(999) # Set seed  
for (i in 1:B) { # Bootstrap sample  
 data\_matrix[i,] <- sample(random\_sample, size = sample\_size, replace = TRUE)  
 theta\_matrix[i] <- mean(data\_matrix[i,])  
}  
hist(theta\_matrix, main = paste('Histogram of the Sample Mean (n=',sample\_size,')'),  
 xlab = 'Bootstrap Sample Mean',  
 sub = paste('Original Sample Mean:', round(sample\_mean, 4),  
 ' Mean Bootstrap Sample Mean:', round(mean(theta\_matrix), 4)))  
mu\_bar\_star <- mean(theta\_matrix)  
mu\_bar\_star - sample\_mean  
2 \* sample\_mean - mu\_bar\_star  
  
(1 / (B - 1)) \* sum((theta\_matrix - mu\_bar\_star)^2)  
  
# part (c)  
concat\_B <- rep(random\_sample, B) # create n\*B vector  
set.seed(999)  
balanced\_bootstrap <- sample(concat\_B, # permute n\*B vector  
 size = length(concat\_B), replace = FALSE)  
# Reference: https://stackoverflow.com/questions/3318333/split-a-vector-into-chunks-in-r  
balanced\_groups <- split(balanced\_bootstrap, # split into B groups  
 rep\_len(1:B, length(balanced\_bootstrap)))  
balanced\_theta <- unlist(lapply(balanced\_groups, mean)) # calculate theta for each  
names(balanced\_theta) <- NULL  
  
hist(balanced\_theta, main = paste('Histogram of the Balanced Sample Mean (n=',sample\_size,')'),  
 xlab = 'Balanced Bootstrap Sample Mean',  
 sub = paste('Original Sample Mean:', round(sample\_mean, 4),  
 ' Mean Bootstrap Sample Mean:', round(mean(balanced\_theta), 4)))  
mu\_bar\_star <- mean(balanced\_theta)  
mu\_bar\_star - sample\_mean  
2 \* sample\_mean - mu\_bar\_star  
(1 / (B - 1)) \* sum((theta\_matrix - mu\_bar\_star)^2)