5.1

Consider the samples 1-6. Use a six-sided die to obtain three different bootstrap samples and their corresponding means.

```
pop <- seq(from = 1, to = 6, by = 1)

n <- 6

s1 <- mean( sample(pop, n, replace = T) )
s2 <- mean( sample(pop, n, replace = T) )
s3 <- mean( sample(pop, n, replace = T) )
```

$\bar{x}_1^* = 3.6666667, \bar{x}_2^* = 3.1666667, \bar{x}_3^* = 4.3333333$

5.2

Consider the samples 1, 3, 4, and 6 from some distribution.

```
pop <- c(1, 3, 4, 6)

samples <- permutations(n = 4, r = 4, pop, repeats.allowed = T)
```

a.) For one random bootstrap sample, find the probability that the mean is one.

```
means <- apply(samples, 1, mean)

p <- mean( means == 1 )
```

Probability: **0.39%**

b.) For one random bootstrap sample, find the probability that the maximum is 6.

```
maxes <- apply(samples, 1, max)

p <- mean( maxes == 6 )
```

Probability: **68.36%**

c.) For one random bootstrap sample, find the probability that exactly two elements in the sample are less than 2.

```
lt2 <- apply(t(apply(samples, 1, function(x) { x < 2})), 1, sum)

p <- mean( lt2 == 2 )
```

Probability: **21.09%**

5.3

Consider the sample 1-3.

a.) List all the (ordered) bootstrap samples from this sample. How many are there?

```
samples <- permutations(n = 3, r = 3, 1:3, repeats.allowed = T)

n <- nrow(samples)
```

Samples: = 3^3 = **27**

b.) How many unordered bootstrap samples are there? For example, {1, 2, 2} and {2, 1, 2} are considered to be the same.

```
samples <- combinations(n = 3, r = 3, 1:3, repeats.allowed = T)

n <- nrow(samples)

assertthat::are_equal(n, choose(3 + 3 - 1, 3))
```

```
[1] TRUE
```

Samples: = $\binom{5}{3}$ = **10**

c.) How many ordered bootstrap samples have one occurrence of 1 and two occurences of 3?

```
samples <- permutations(n = 3, r = 3, 1:3, repeats.allowed = T)

n.ones <- apply(t(apply(samples, 1, FUN = function(x) { x == 1 })), 1, function(x) sum(x) )
n.threes <- apply(t(apply(samples, 1, FUN = function(x) { x == 3 })), 1, function(x) sum(x) )

sum((n.ones == 1 & n.threes == 2) == T)
```

```
[1] 3
```

Is this the same number of bootstrap samples that have each of 1, 2 and 3 occuring exactly once?

```
n.ones <- apply(t(apply(samples, 1, FUN = function(x) { x == 1 })), 1, function(x) sum(x) )
n.twos <- apply(t(apply(samples, 1, FUN = function(x) { x == 2 })), 1, function(x) sum(x) )
n.threes <- apply(t(apply(samples, 1, FUN = function(x) { x == 3 })), 1, function(x) sum(x) )

sum((n.ones == 1 & n.twos == 1 & n.threes == 1) == T)
```

```
[1] 6
```

No, 3 != 6.

d.) Is the probability of obtaining a bootstrap sample with one 1 and two 3's the same as the probabiliity of obtaining a bootstrap sample with each of 1, 2 and 3 occuring exactly once?

```
( sum((n.ones == 1 & n.threes == 2)) / n ) == ( sum((n.ones == 1 & n.twos == 1 &  n.threes == 1
```

```
[1] FALSE
```

No, 3% and 6% chances respectfully.

5.4

Consider the samples 1, 3, 3, and 5 from some distribution.

```
samples <- c(1, 3, 3, 5)
```

a.) How many bootstrap samples are there?

```
boot <- permutations(n = 3, r = 4, v = samples, repeats.allowed = T)

n <- nrow(boot)
```

3 unique items to pick from, 4 places to put each item.

Number of permutations: 3^4 = *81*

b.) List the distinct bootstrap samples assuming order does not matter.

```
combinations(n = 3, r = 4, v = samples, repeats.allowed = T)
```

```
      [,1] [,2] [,3] [,4]
 [1,]    1    1    1    1
 [2,]    1    1    1    3
 [3,]    1    1    1    5
 [4,]    1    1    3    3
 [5,]    1    1    3    5
 [6,]    1    1    5    5
 [7,]    1    3    3    3
 [8,]    1    3    3    5
 [9,]    1    3    5    5
[10,]    1    5    5    5
[11,]    3    3    3    3
[12,]    3    3    3    5
[13,]    3    3    5    5
[14,]    3    5    5    5
[15,]    5    5    5    5
```

```
choose(4 + 3 - 1, 4)
```

```
[1] 15
```

5.5

We determine the number of distinct bootstrap samples from a finite set.

a.) A bakery sells five types of cookies: sugar, chocolate chip, oatmeal, peanut butter, and ginger snap. Show that the number of ways to order 5 cookies is $\binom{9}{5}$

Unordered samping with replacement: $\binom{n+k-1}{k}, n = 5, k = 5$

```
choose(5 + 5 - 1, 5)
```

```
[1] 126
```

b.) Show that the number of sets of size n (order does not matter) drawn with replacement from the (distinct) $a_1, a_2, \dots, a_n$ is $\binom{2n-1}{n}$

Conclude that the number of distinct bootstrap samples from the set $[a_1, a_2, \dots, a_n]$ is $\binom{2n-1}{n}$

5.6

Let $k_1, k_2, \dots, k_n$ denote non-negative integers satisfying $k_1 + k_2 + \dots + k_n = n$, and suppose the elements in the set $a_1, a_2, \dots, a_n$ are distinct.

a.) Show that the number of bootstrap samples with k_1 occurrences of a_1, k_2 occurrences of $a_2, \dots, k_n$ occurrences of a_n is $\binom{n}{k_1, k_2, \dots, k_n}$

b.) Compute the probability that a randomly drawn bootstrap sample will have k_i occurrences of $a_i, i = 1, 2, \dots, n$

5.7

Refer to Example 5.4 and the remark at the end of the example.

a.) What might account for the fact that there were more missing values for the men who skateboarded in front of the male experimenter? How might this bias the outcome?

It could be that the approached skateboarders refused to participate in the study of performing tricks in front of other men.

b.) Why do you suppose it was important that the two experimenters were blinded to the purpose of the study?

The female could have flurted or otherwise influenced skateboarders who were performing tricks if they knew the intent of the study.

5.8

Consider a population that has a normal distribution with mean $\mu = 36$, standard deviation $\sigma = 8$.

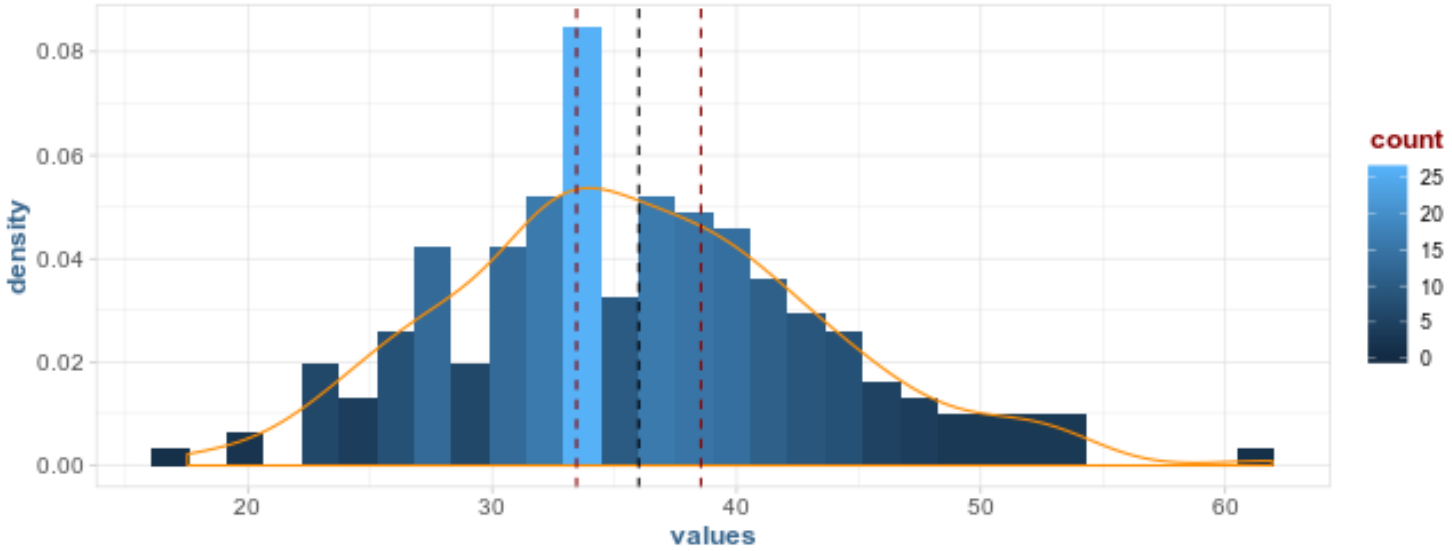
```
mu <- 36; sigma <- 8; n <- 200

se <- mu / sqrt(n)
```

a.) The sampling distribution of $\bar{X}$ for samples of size 200 will have what mean, standard error, and shape?

$\mu = 36$, *SE = 36 / sqrt(200) = 2.5455844, shape will be approximately normal (CLT).*

b.) Use R to draw a random sample of size 200 from this population. Conduct EDA on your sample.

```
set.seed(123)

samp <- data.table(values = rnorm(200, mean = 36, sd = 8))

ggplot(samp, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), color = "darkorange") +
   geom_vline(xintercept = mu, col = "black", lty = 2) +
   geom_vline(xintercept = mu - se, col = "darkred", lty = 2) +
   geom_vline(xintercept = mu + se, col = "darkred", lty = 2)
```



c.) Compute the bootstrap distribution for your sample, and note the bootstrap mean and standard error.

```
boot.fn <- function(data, index){
   mean(data[index]$values)
}

I <- 10e3

boot(samp, boot.fn, R = I) # boot pkg
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
```

```
boot(data = samp, statistic = boot.fn, R = I)


Bootstrap Statistics :
    original      bias    std. error
t1* 35.93144 0.006716095   0.5392066
```

```
cst.boot <- function(values, n, I = 10e3, alpha = 0.05) {
   bootstrap <- numeric(I)

   for(i in 1:I)
   {
      bootstrap[i] <- mean( sample(values, n, replace = T) )
   }

   observed <- mean(values)

   boot.mean <- mean(bootstrap)
   boot.bias <- observed - boot.mean
   boot.se <- sd(bootstrap)

   list(bootstrap = bootstrap,
         observed = observed,
         mean = boot.mean,
         bias = boot.bias,
         se = boot.se,
         conf = quantile(bootstrap, c(alpha/2, 1 - alpha/2)))
}
```

d.) Compare the bootstrap distribution to the theoretical sampling distribution by creating a table like Table 5.2:

```
n.200 <- cst.boot(samp$values, 200)

tbl <- data.table(Data = c("Population", "Sampling Distribution", "Sample", "Bootstrap Distribu
                  Mean = c(mu, mu, n.200$observed, n.200$mean),
                  SD = c(sigma, mu/sqrt(n), sd(samp$values), n.200$se))

pretty_kable(tbl, "Sampling Statistics")
```

Table 1: Sampling Statistics

Data	Mean	SD
Population	36.00	8.00
Sampling Distribution	36.00	2.55
Sample	35.93	7.55
Bootstrap Distribution	35.94	0.53

e.) Repeat for sample sizes $n = 50$ and $n = 10$. Carefully describe yyour observations about the effects of sample size on the bootstrap distribuiton.

```
n <- 50
samp <- data.table(values = rnorm(n, mean = mu, sd = sigma))
n.50 <- cst.boot(samp$values, n)

tbl <- data.table(Data = c("Population", "Sampling Distribution", "Sample", "Bootstrap Distribu
                  Mean = c(mu, sigma, n.50$observed, n.50$mean),
                  SD = c(sigma, mu/sqrt(n), sd(samp$values), n.50$se))

pretty_kable(tbl, "Sampling Statistics")
```

Table 2: Sampling Statistics

Data	Mean	SD
Population	36.00	8.00
Sampling Distribution	8.00	5.09
Sample	36.94	7.78
Bootstrap Distribution	36.96	1.09

```
n <- 10
samp <- data.table(values = rnorm(n, mean = mu, sd = sigma))
n.10 <- cst.boot(samp$values, 10)

tbl <- data.table(Data = c("Population", "Sampling Distribution", "Sample", "Bootstrap Distribu
                  Mean = c(mu, sigma, n.10$observed, n.10$mean),
                  SD = c(sigma, mu/sqrt(n), sd(samp$values), n.10$se))

pretty_kable(tbl, "Sampling Statistics")
```

Table 3: Sampling Statistics

Data	Mean	SD
Population	36.00	8.00
Sampling Distribution	8.00	11.38
Sample	32.42	10.58
Bootstrap Distribution	32.40	3.15

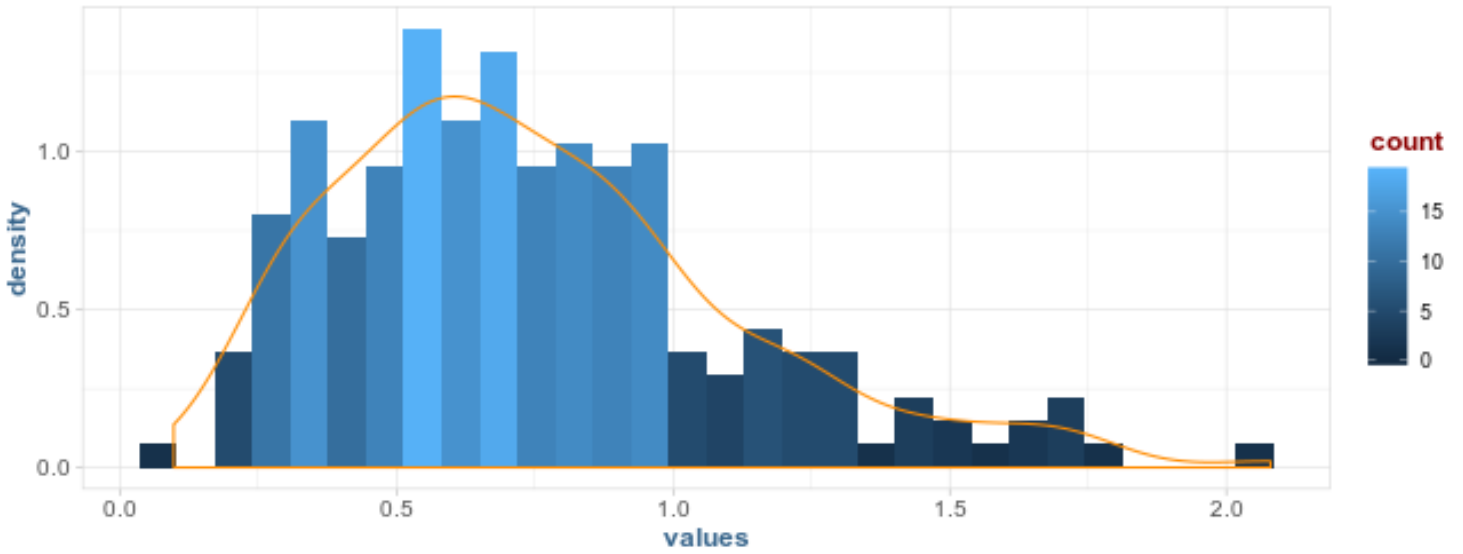
The center of the bootstrap distribution doesn't vary much with smaller n, however, confidence intervals (the sd of the bootstrap dist) vary wildy.

5.9

Consider a population that has a gamma distribution with parameters r = 5, $\lambda = 4$.

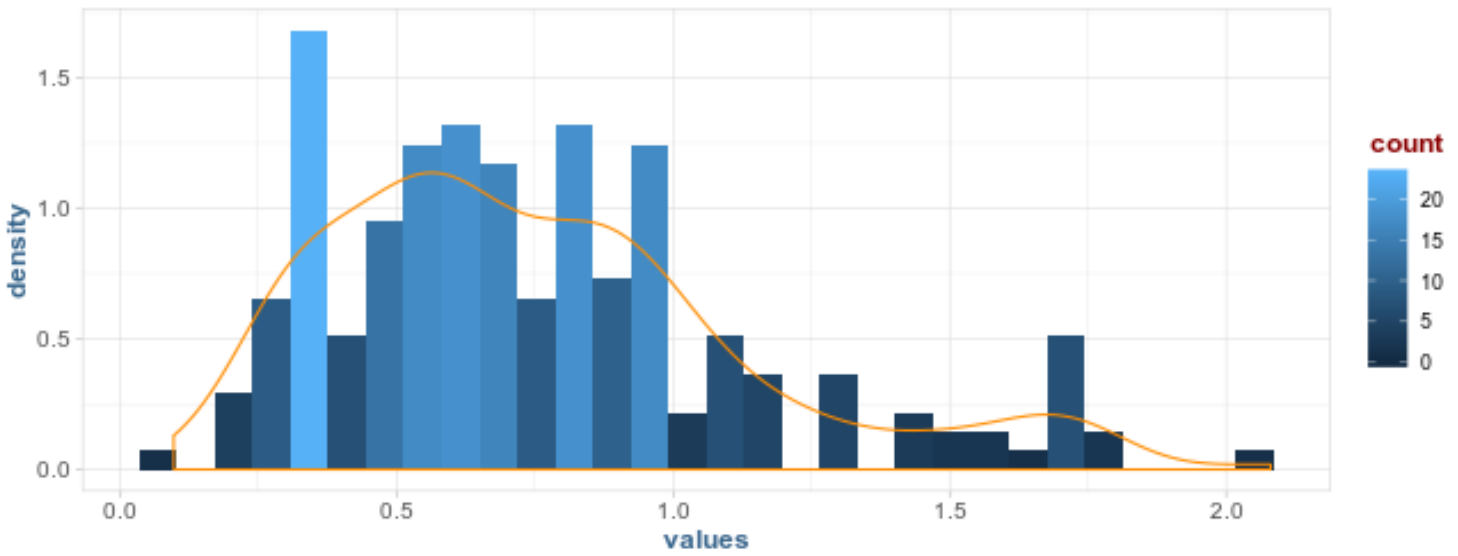
a.) Use simulation (with $n = 200$) to generate an approximate sampling distribution of the mean; plot and describe the distribution.

```
set.seed(123)

n <- 200; r <- 5; lambda <- 4; mu <- lambda/r

pop <- data.table(values = rgamma(n, shape = lambda, rate = r))

ggplot(pop, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange")
```



b.) Now, draw one random sample of size 200 from this population. Create a histogram of your sample, and find the mean and standard deviation.

```
samp <- data.table(values = sample(pop$values, n, replace = T))

ggplot(samp, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange")
```




```
xbar <- mean(samp$values); sd <- sd(samp$values)

xbar; sd
```

```
[1] 0.7571211
```

```
[1] 0.3859094
```

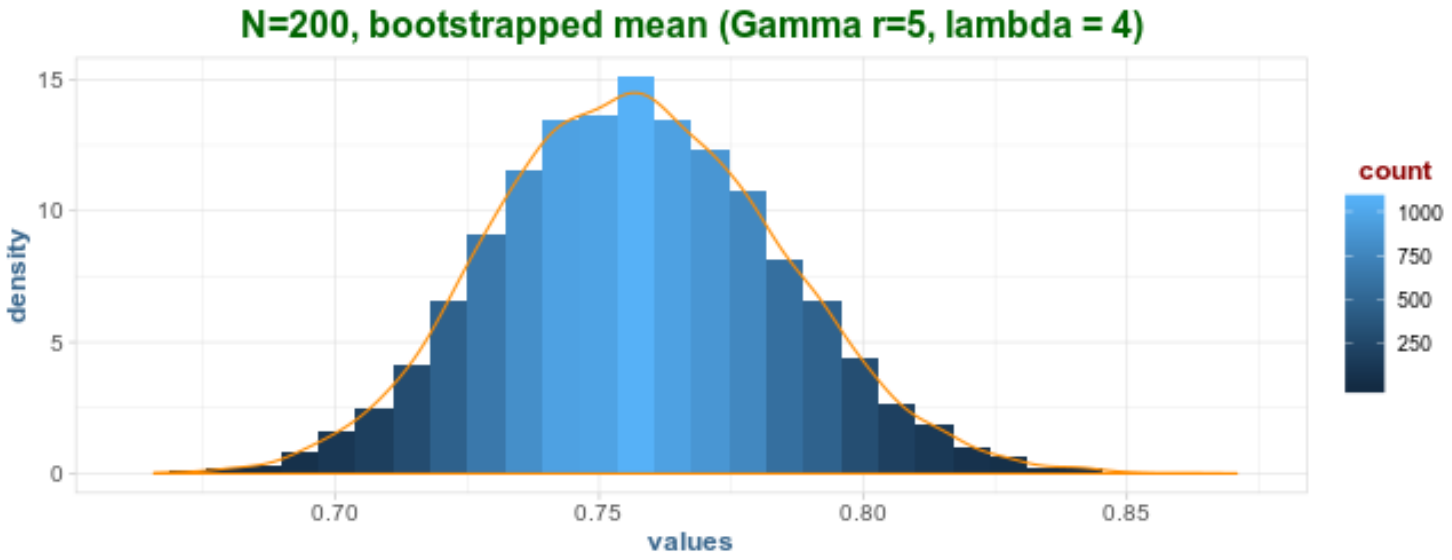
c.) Compute the bootstrap distribution of the mean for you sample, plot it, and note the bootstrap mean and standard error.

```
I <- 10e3

boot.fn <- function(data, index) {
   mean(data[index]$values)
}

boot(samp, boot.fn, R = I)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = samp, statistic = boot.fn, R = I)


Bootstrap Statistics :
     original       bias    std. error
t1* 0.7571211 0.0003439594  0.02732601
```

```
n.200 <- cst.boot(samp$values, n)
```

```
ggplot(data.table(values = n.200$bootstrap), aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange") +
   labs(title = "N=200, bootstrapped mean (Gamma r=5, lambda = 4)")
```



d.) Compare the bootstrap distribution to the approximate theoretical sampling distribution by creating a table like Table 5.2.

```
tbl <- data.table(Data = c("Population", "Sampling Distribution", "Sample", "Bootstrap Distribu
                  Mean = c(mu, sd(samp$values), n.200$observed, n.200$mean),
                  SD = c(mu, mu/sqrt(n), sd(samp$values), n.200$se))

pretty_kable(tbl, "Sampling Statistics")
```

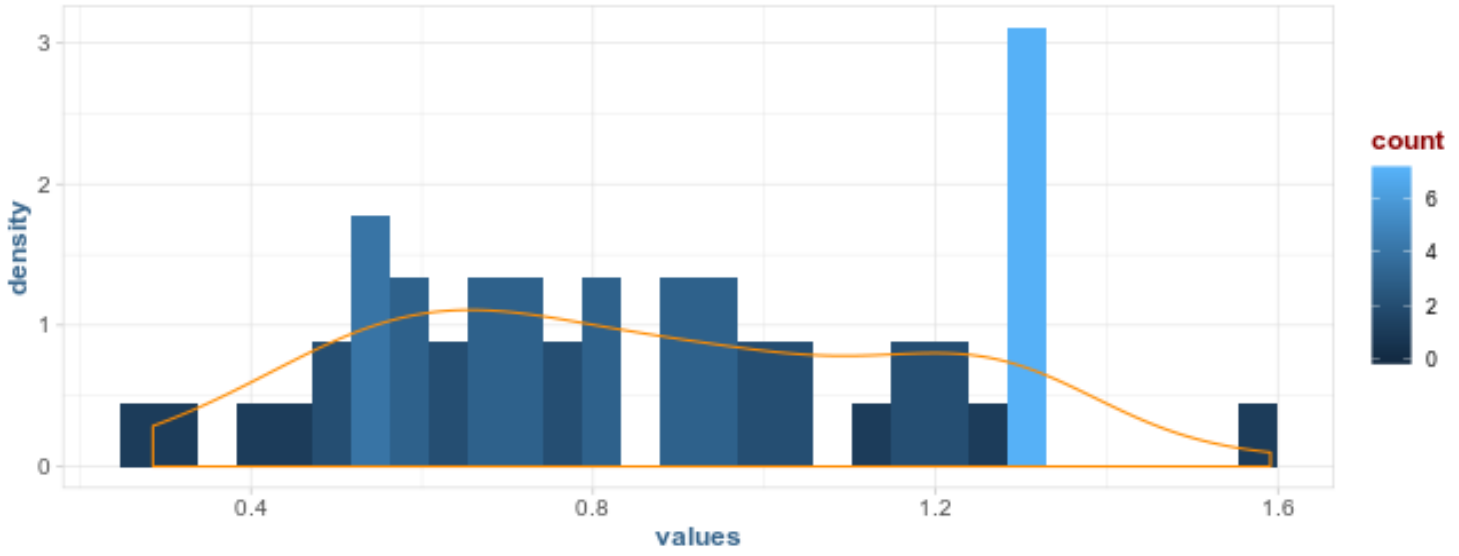
Table 4: Sampling Statistics

Data	Mean	SD
Population	0.80	0.80
Sampling Distribution	0.39	0.06
Sample	0.76	0.39
Bootstrap Distribution	0.76	0.03

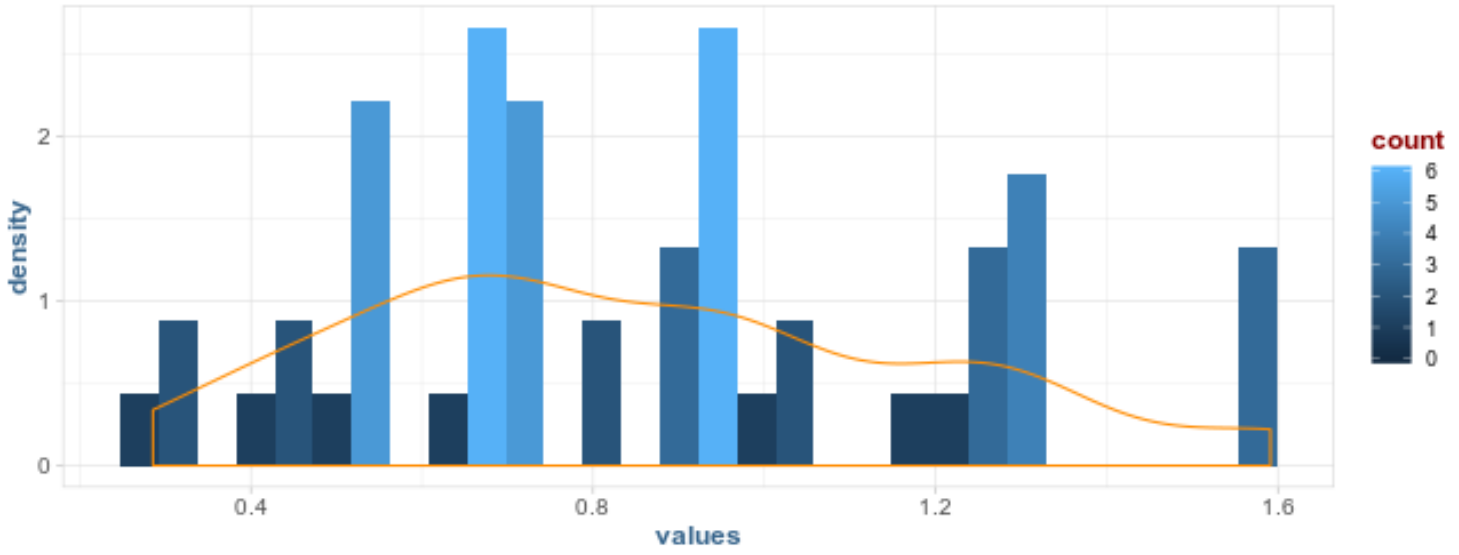
e.) Repeat (a-e) for sample sizes of n = 50, and n = 10. Describe carefully your observations about the effects of sample size on the bootstrap distribution.

```
n <- 50

pop <- data.table(values = rgamma(n, shape = lambda, rate = r))

ggplot(pop, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
```

```
   geom_density(aes(values), col = "darkorange")
```




```
samp <- data.table(values = sample(pop$values, n, replace = T))

ggplot(samp, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange")
```

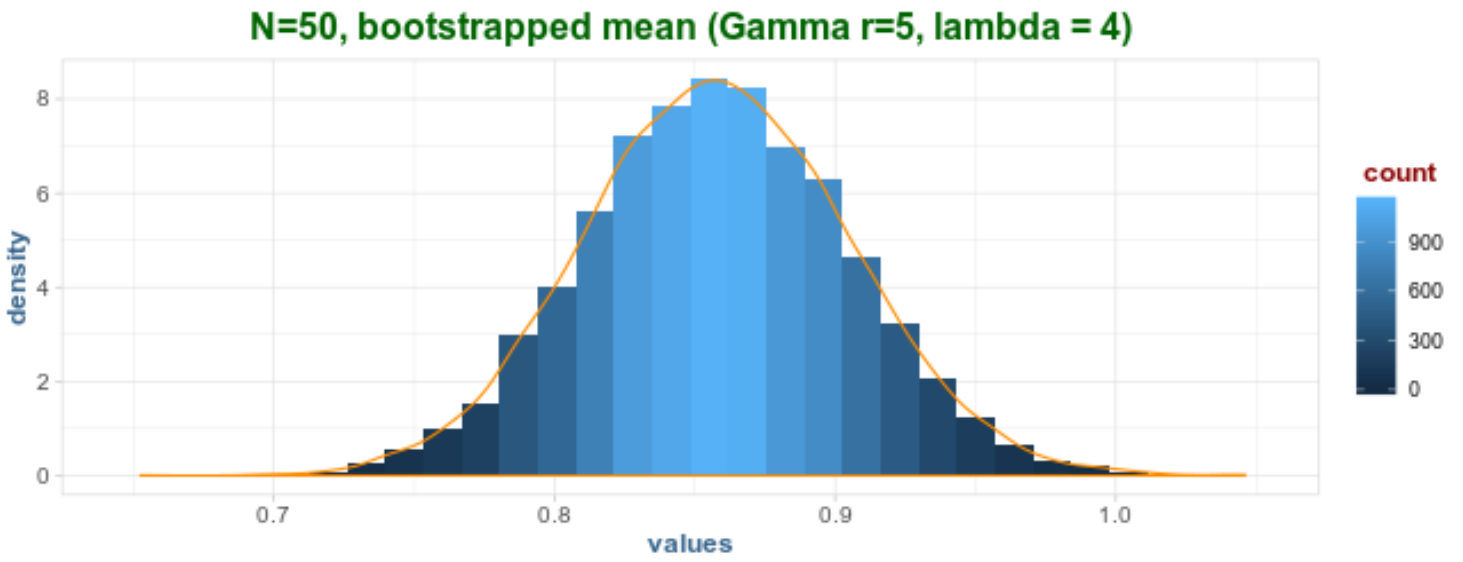



```
xbar <- mean(samp$values); sd <- sd(samp$values)

xbar; sd
```

```
[1] 0.8573007
```

```
[1] 0.3391036
```

```
n.50 <- cst.boot(samp$values, n)

ggplot(data.table(values = n.50$bootstrap), aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange") +
   labs(title = "N=50, bootstrapped mean (Gamma r=5, lambda = 4)")
```

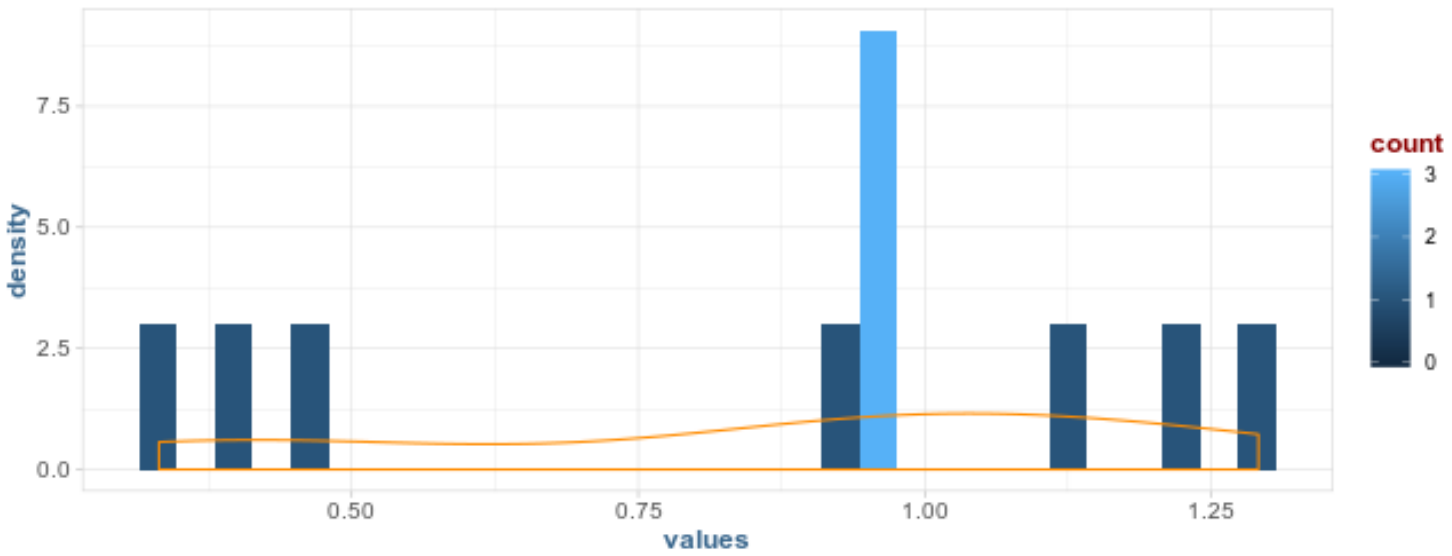



```
tbl <- data.table(Data = c("Population", "Sampling Distribution", "Sample", "Bootstrap Distribu
                  Mean = c(mu, sd(samp$values), n.50$observed, n.50$mean),
                  SD = c(mu, mu/sqrt(n), sd(samp$values), n.50$se))

pretty_kable(tbl, "Sampling Statistics")
```

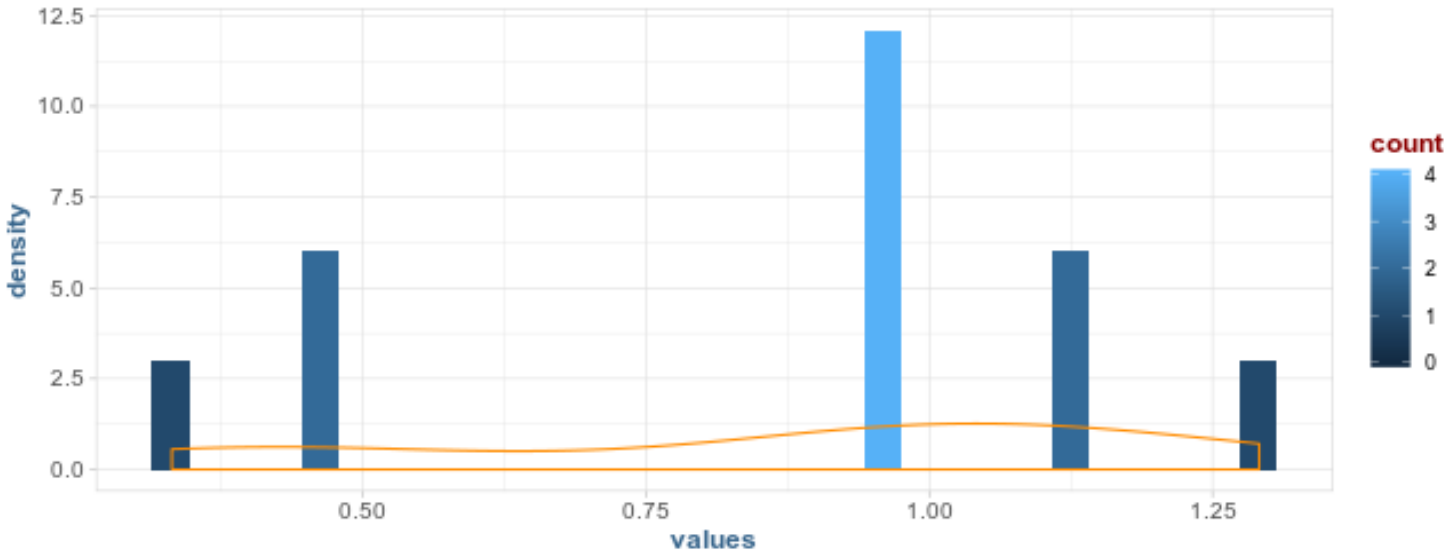
Table 5: Sampling Statistics

Data	Mean	SD
Population	0.80	0.80
Sampling Distribution	0.34	0.11
Sample	0.86	0.34
Bootstrap Distribution	0.86	0.05

```
n <- 10

pop <- data.table(values = rgamma(n, shape = lambda, rate = r))

ggplot(pop, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange")
```




```
samp <- data.table(values = sample(pop$values, n, replace = T))

ggplot(samp, aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange")
```

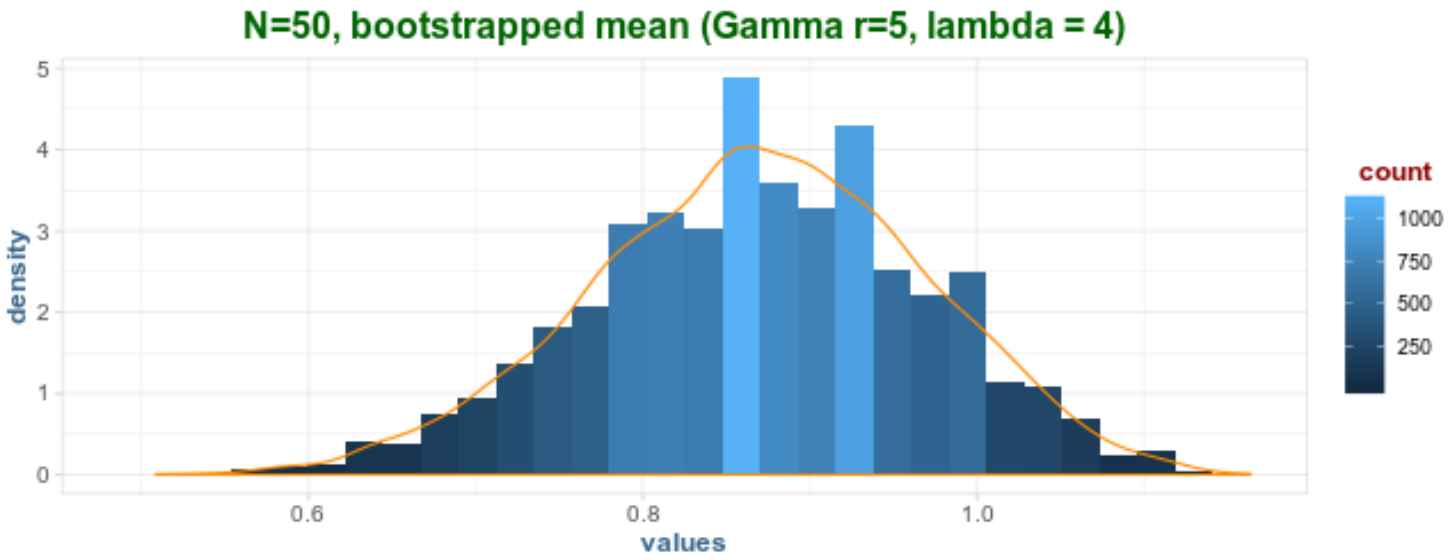



```
xbar <- mean(samp$values); sd <- sd(samp$values)

xbar; sd
```

```
[1] 0.8682445
```

```
[1] 0.3325161
```

```
n.50 <- cst.boot(samp$values, n)

ggplot(data.table(values = n.50$bootstrap), aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange") +
   labs(title = "N=50, bootstrapped mean (Gamma r=5, lambda = 4)")
```




```
tbl <- data.table(Data = c("Population", "Sampling Distribution", "Sample", "Bootstrap Distribu
                  Mean = c(mu, sd(samp$values), n.50$observed, n.50$mean),
                  SD = c(mu, mu/sqrt(n), sd(samp$values), n.50$se))

pretty_kable(tbl, "Sampling Statistics")
```

Table 6: Sampling Statistics

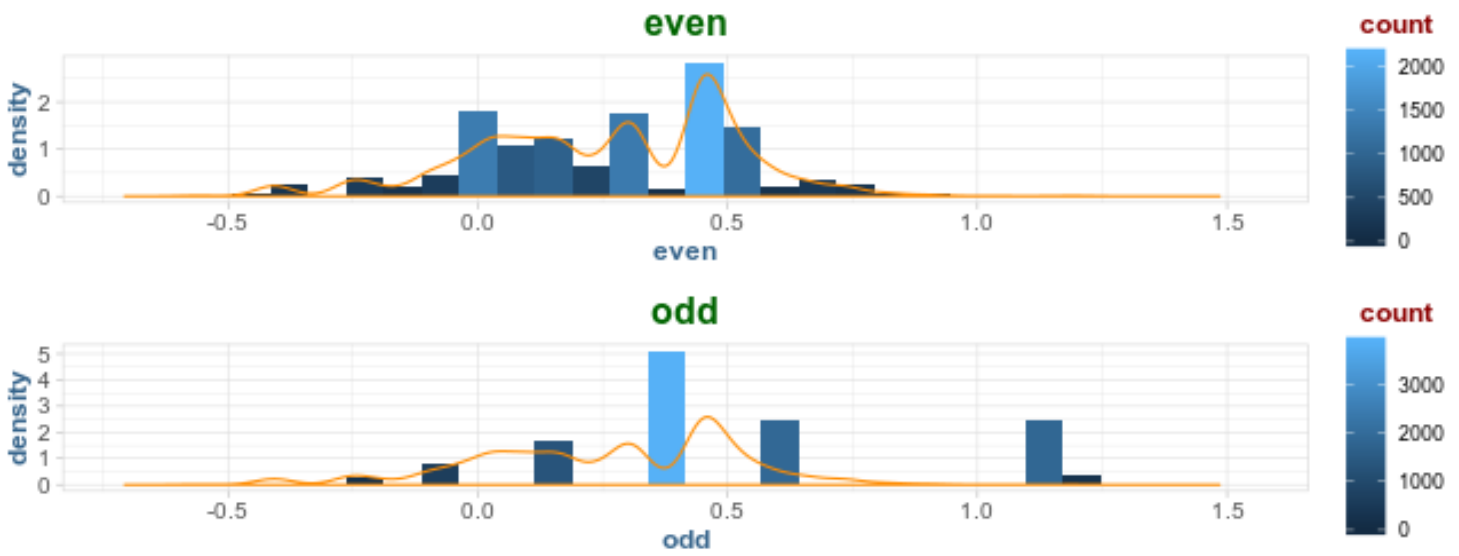
Data	Mean	SD
Population	0.80	0.80
Sampling Distribution	0.33	0.25
Sample	0.87	0.33
Bootstrap Distribution	0.87	0.10

5.10

We investigate the bootstrap distribution of the median. Create random sample of size n for various n and bootstrap the median. Describe the bootstrap distribution.

```
ne <- 14 # n even
no <- 15 # n odd
```

```r
wwe <- rnorm(ne) # draw samples of size ne
wwo <- rnorm(no) # draw random samples of size no

N <- 10^4

even.boot <- numeric(N) # save space
odd.boot <- numeric(N)

for(i in 1:N)
{
   x.even <- sample(wwe, ne, replace = T)
   x.odd <- sample(wwo, no, replace = T)

   even.boot[i] <- median(x.even)
   odd.boot[i] <- median(x.odd)
}

boot <- data.table(even = even.boot, odd = odd.boot)

p1 <- ggplot(boot, aes(even)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange") +
   labs(title = "even")

p2 <- ggplot(boot, aes(odd)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange") +
   labs(title = "odd")

gridExtra::grid.arrange(p1, p2)
```

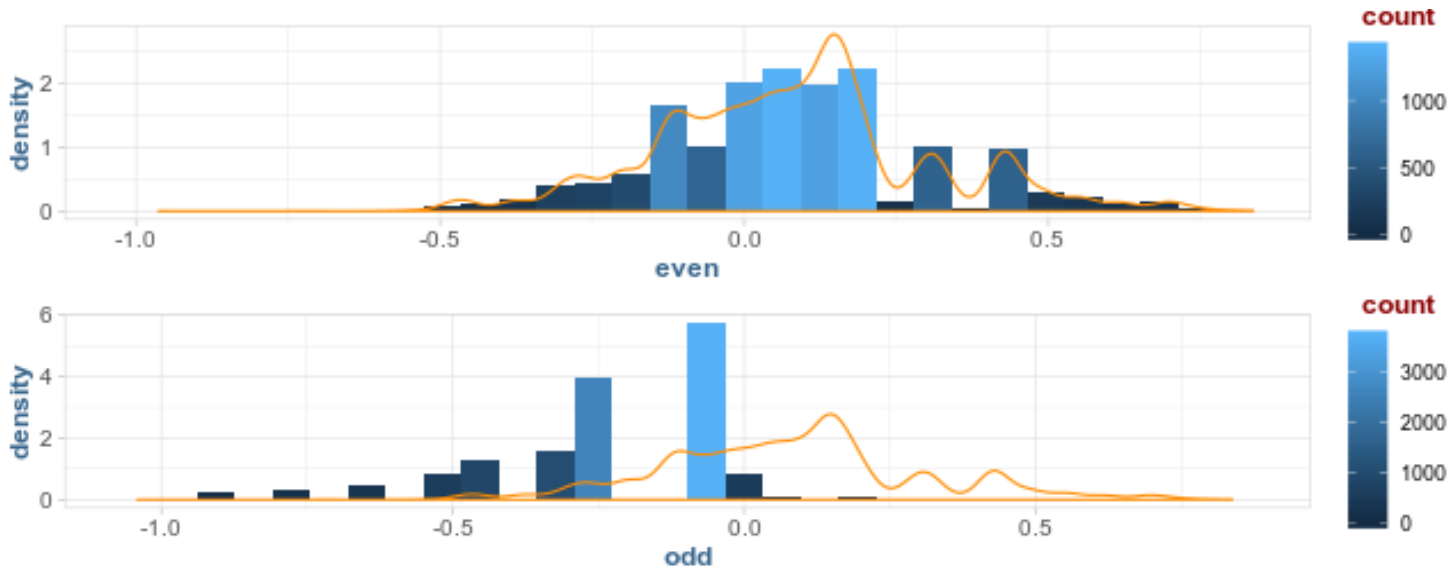


Change the sample sizes to 36 and 37; 200 and 201; and 10,000 and 10,001.

Note the similarities/dissimalarities, trends, and so on. Why does the parity of the sample size matter? (*Note: Adjust the x limits in the plots as needed.*)

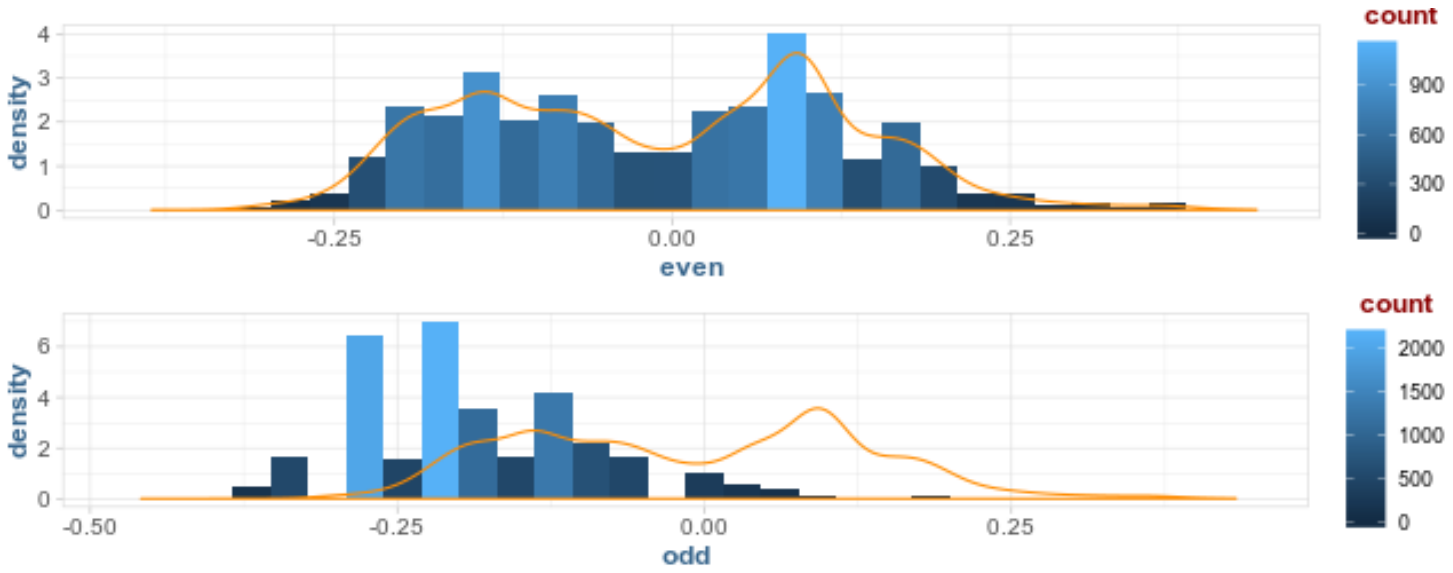
```
ne <- 36 # n even
no <- 37 # n odd

wwe <- rnorm(ne) # draw samples of size ne
wwo <- rnorm(no) # draw random samples of size no

N <- 10^4

even.boot <- numeric(N) # save space
odd.boot <- numeric(N)

for(i in 1:N)
{
   x.even <- sample(wwe, ne, replace = T)
   x.odd <- sample(wwo, no, replace = T)

   even.boot[i] <- median(x.even)
   odd.boot[i] <- median(x.odd)
}

boot <- data.table(even = even.boot, odd = odd.boot)

p1 <- ggplot(boot, aes(even)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange")
```

```
p2 <- ggplot(boot, aes(odd)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange")

gridExtra::grid.arrange(p1, p2)
```



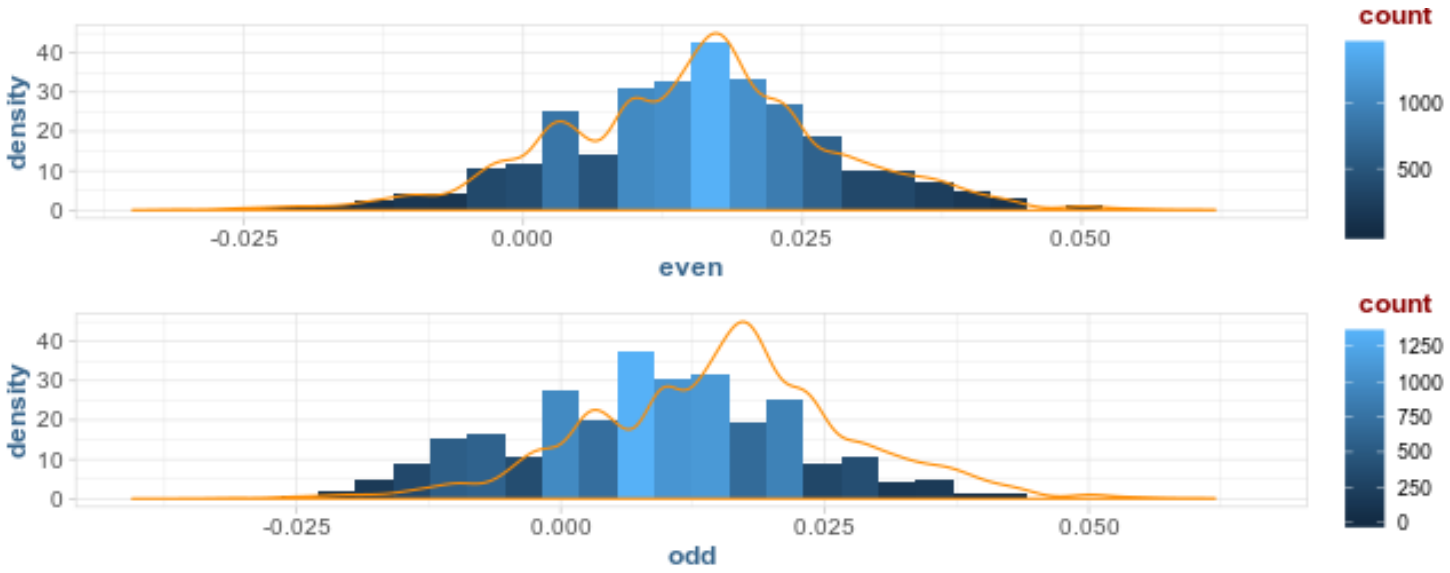

```
ne <- 200 # n even
no <- 201 # n odd

wwe <- rnorm(ne) # draw samples of size ne
wwo <- rnorm(no) # draw random samples of size no

N <- 10^4

even.boot <- numeric(N) # save space
odd.boot <- numeric(N)

for(i in 1:N)
{
   x.even <- sample(wwe, ne, replace = T)
   x.odd <- sample(wwo, no, replace = T)

   even.boot[i] <- median(x.even)
   odd.boot[i] <- median(x.odd)
}

boot <- data.table(even = even.boot, odd = odd.boot)
```

```
p1 <- ggplot(boot, aes(even)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange")

p2 <- ggplot(boot, aes(odd)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange")

gridExtra::grid.arrange(p1, p2)
```




```
ne <- 10000 # n even
no <- 10001 # n odd

wwe <- rnorm(ne) # draw samples of size ne
wwo <- rnorm(no) # draw random samples of size no

N <- 10^4

even.boot <- numeric(N) # save space
odd.boot <- numeric(N)

for(i in 1:N)
{
   x.even <- sample(wwe, ne, replace = T)
   x.odd <- sample(wwo, no, replace = T)

   even.boot[i] <- median(x.even)
   odd.boot[i] <- median(x.odd)
}
```

```
boot <- data.table(even = even.boot, odd = odd.boot)

p1 <- ggplot(boot, aes(even)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange")

p2 <- ggplot(boot, aes(odd)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(even), col = "darkorange")

gridExtra::grid.arrange(p1, p2)
```



For odd n, median will be one of the sample points. For smaller n, there will be only n possible values for the median, so the sampling distribution is more “granular” than when n is even.

5.11

Import the data from data set Bangladesh. In addition to aresnic concentrations for 271 wells, the data set contains cobolt and chlorine concentrations.

a.) Conduct EDA on the chlorine concentrations and describe the salient features.

```
Bangladesh <- data.table(read.csv(paste0(data.dir, "Bangladesh.csv"),
                                  header = T))
head(Bangladesh)
```

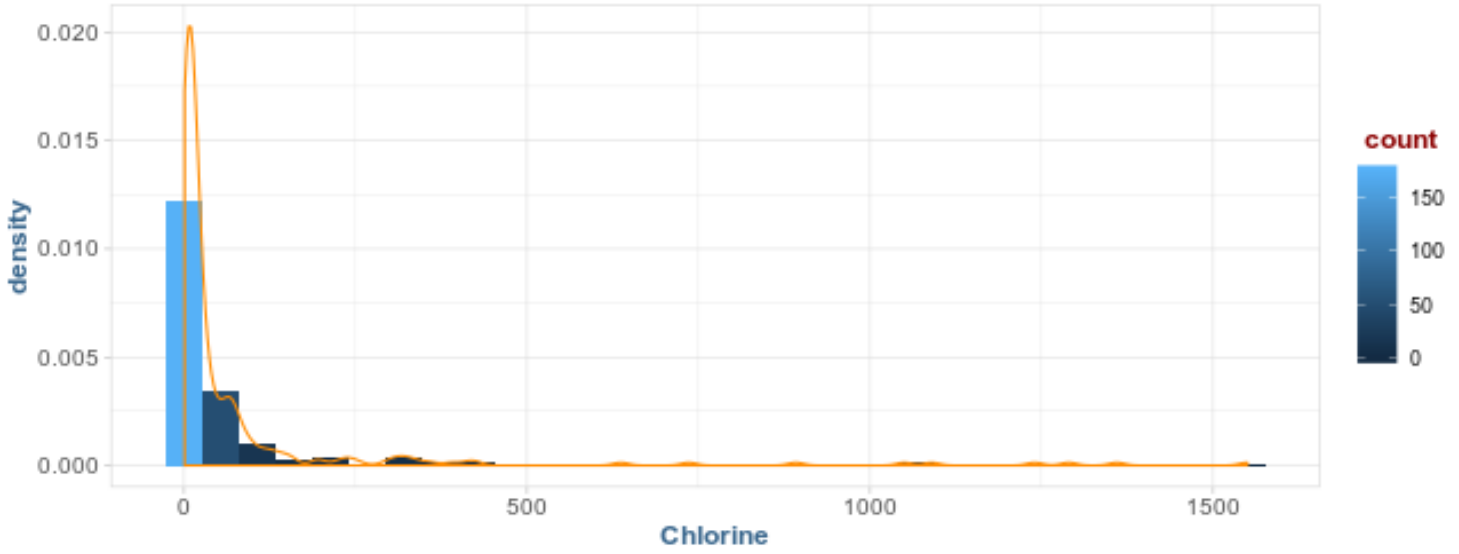
```
   Arsenic Chlorine Cobalt
1:    2400      6.2   0.42
2:       6    116.0   0.45
```

```
3:     904     14.8   0.63
4:     321     35.9   0.68
5:    1280     18.9   0.58
6:     151      7.8   0.35
```

```
ggplot(Bangladesh, aes(Chlorine)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(Chlorine), col = "darkorange")
```

```
Warning: Removed 2 rows containing non-finite values (stat_bin).
```

```
Warning: Removed 2 rows containing non-finite values (stat_density).
```




```
GGally::ggpairs(Bangladesh)
```

```
Registered S3 method overwritten by 'GGally':
  method from
  +.gg   ggplot2
```

```
Warning in (function (data, mapping, alignPercent = 0.6, method = "pearson", :
Removed 2 rows containing missing values
```

```
Warning in (function (data, mapping, alignPercent = 0.6, method = "pearson", :
Removing 1 row that contained a missing value
```

```
Warning: Removed 2 rows containing missing values (geom_point).
```

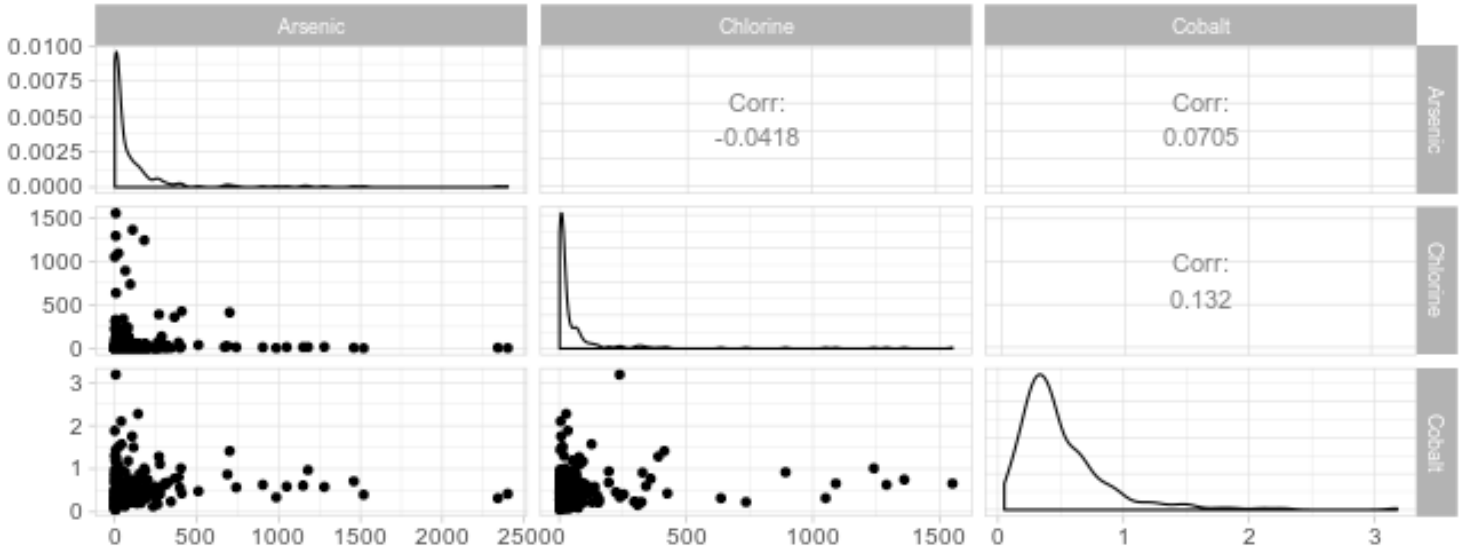
```
Warning: Removed 2 rows containing non-finite values (stat_density).
```

```
Warning in (function (data, mapping, alignPercent = 0.6, method = "pearson", :
Removed 3 rows containing missing values
```

```
Warning: Removed 1 rows containing missing values (geom_point).
```

```
Warning: Removed 3 rows containing missing values (geom_point).
```

```
Warning: Removed 1 rows containing non-finite values (stat_density).
```



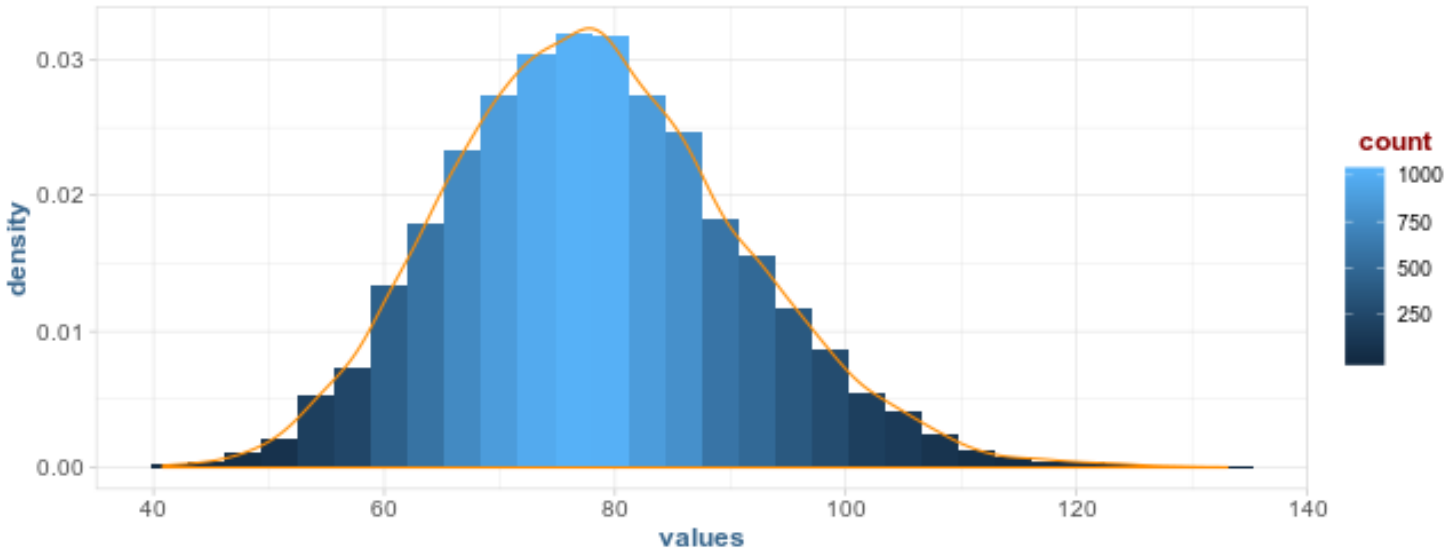
b.) Bootstrap the mean.

```
N <- 10e3

boot.fn <- function(data, index){
   mean(data[index], na.rm = T)
}

boot(Bangladesh$Chlorine, boot.fn, R = N)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Bangladesh$Chlorine, statistic = boot.fn, R = N)


Bootstrap Statistics :
    original     bias    std. error
t1* 78.08401 0.04234321    12.72407
```

```
observed <- mean(Bangladesh$Chlorine, na.rm = T)

bootstrap <- numeric(N)

for(i in 1:N)
```

```
{
   bootstrap[i] <- mean(sample(Bangladesh$Chlorine, size = nrow(Bangladesh), replace = T), na.r
}

ggplot(data.table(values = bootstrap), aes(values)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(aes(values), col = "darkorange")
```



c.) Find and interpret the 95% bootstrap percentile confidence interval.

```
alpha <- 0.05
quantile(bootstrap, c(alpha/2, 1 - alpha/2))
```

```
     2.5%     97.5%
 55.13463 104.69678
```

The spread on the confidence interval is extremely large, which is unsuprising given the heavly skewed distribtion of the sample.

d.) What is the bootstrap estimate of the bias? What fraction of the bootstrap standard error does it represent?

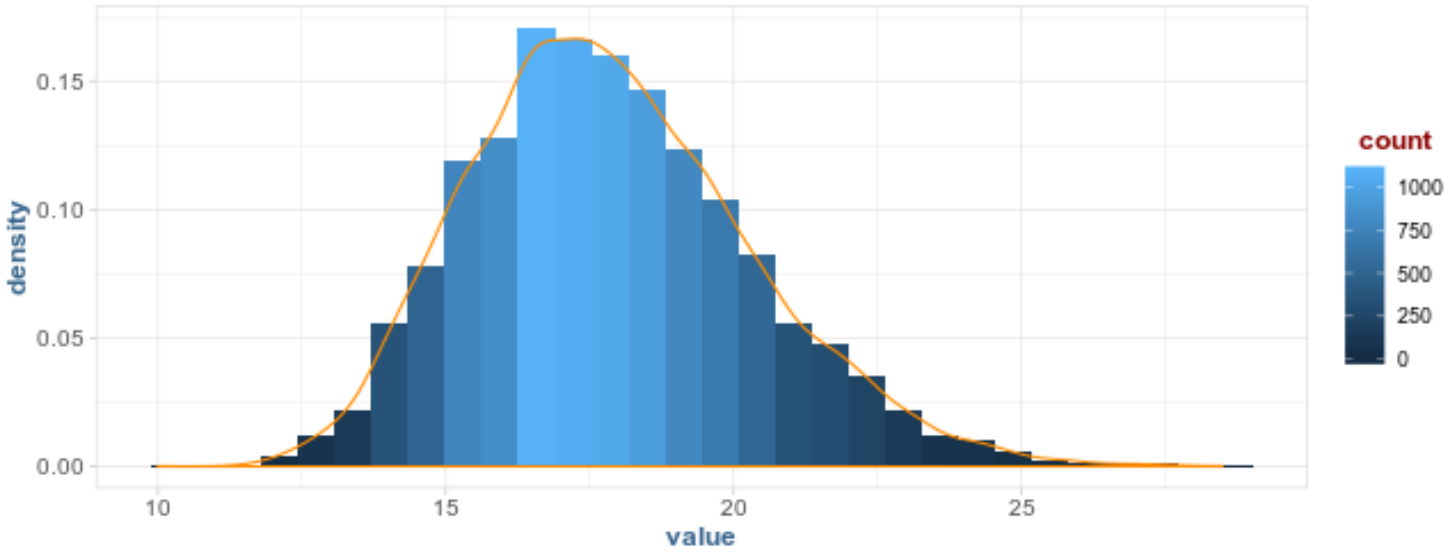
```
bias <- observed - mean(bootstrap)

sd(bootstrap) / bias
```

```
[1] -88.94388
```

5.12

Consider Bangladesh chlorine (concentration). Bootstrap the trimmed mean (say, trim the upper and lower 25%), and compare your results with the usual mean (previous result).

```
values <- Bangladesh[!is.na(Chlorine)]$Chlorine

n <- length(values); N <- 10e3; trim <- .25

observed <- mean(values, trim = trim)
bootstrap <- vector(mode = "numeric", length = N)

for(i in 1:N)
{
   bootstrap[i] <- mean( sample(values, n, replace = T), trim = trim )
}

ggplot(data.table(value = bootstrap), aes(value)) +
   geom_histogram(aes(y = ..density.., fill = ..count..), bins = 30) +
   geom_density(col = "darkorange")
```




```
alpha <- 0.05

boot.mean <- mean(bootstrap)
boot.bias <- observed - boot.mean
boot.se <- sd(bootstrap)

quantile(bootstrap, c(alpha/2, 1 - alpha/2))
```

```
    2.5%    97.5%
13.70583 23.14009
```

```
# boot pkg

boot.fn <- function(data, index){
```

```
   mean( data[index], trim = trim)
}

boot(values, boot.fn, R = N)
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = values, statistic = boot.fn, R = N)


Bootstrap Statistics :
    original    bias    std. error
t1*  17.6363 0.2632425    2.469784
```