Resampling

Contents

- We will use the following throughout the slide-set.

```r
sharks <- read.csv("../Data/Sharks/sharks.csv")
popSharks <- rownames(sharks)

# A function to put n or N in the denominator of SD
# rather than n-1 or N-1
sdn <- function( y.pop ) {
  N = length(y.pop)
  sqrt( var(y.pop)*(N-1)/(N)  )    }


#samples <- combn(popSharks, 5)
#N_s <-  ncol(samples)
```

```
N_s <- 10^4
n = 6
set.seed(341)
samples <- sapply(1:N_s, FUN =function(b)  sample(popSharks, n, replace = TRUE) )

avePop <- mean(sharks[, "Length"])
avesSamp <- apply(samples, MARGIN = 2,
                  FUN = function(s){mean(sharks[s,"Length"])})
sampleErrors <- avesSamp - avePop

tmpAve <- mean(avesSamp)
tmpSD <- sdn(avesSamp)

sdsSamp <- apply(samples, MARGIN = 2,
                  FUN = function(s){sdn(sharks[s,"Length"])})
```

4.3 Resampling

- As shown in previous sections, understanding the sampling behaviour of sample attributes is essential for making inferences about any population attribute. e.g.
  - For a discrepancy measure whose sampling distribution allows us to test hypotheses and
  - a pivotal quantity whose sampling distribution allows us to construct confidence intervals.

- However, this process requires undertaking repeated sampling from the population

- In practice, however,
  - the population from which our sample was taken cannot be repeatedly sampled for our purposes,
  - we have only one sample.

Recall - Resampling from the Population

- Draw sample $\mathcal{S}$ of size n from a study population $\mathcal{P}$ according to some sampling mechanism
  - then calculate the sample attribute $a(\mathcal{S})$ to estimate its population counterpart $a(\mathcal{P})$.

- To understand the sampling distribution of any attribute $a(\mathcal{S})$,
  - we draw M samples $\mathcal{S}_1, \ldots, \mathcal{S}_M$ and

- use the values $a(\mathcal{S}_1), \ldots, a(\mathcal{S}_M)$ to inform us about the sampling distribution of $a(\mathcal{S})$.

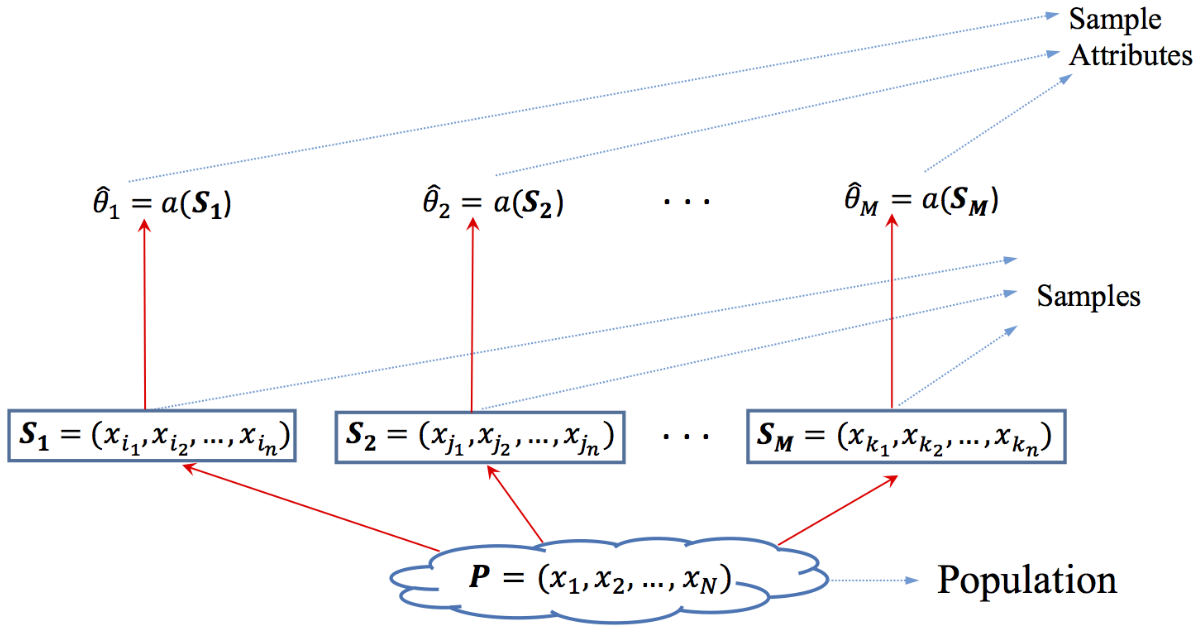

Sample Attributes
$\hat{\theta}_1 = a(\boldsymbol{S}_1)$
$\hat{\theta}_2 = a(\boldsymbol{S}_2)$
$\cdots$
$\hat{\theta}_M = a(\boldsymbol{S}_M)$
Samples
$\boldsymbol{S}_1 = (x_{i_1}, x_{i_2}, \ldots, x_{i_n})$
$\boldsymbol{S}_2 = (x_{j_1}, x_{j_2}, \ldots, x_{j_n})$
$\cdots$
$\boldsymbol{S}_M = (x_{k_1}, x_{k_2}, \ldots, x_{k_n})$
$\boldsymbol{P} = (x_1, x_2, \ldots, x_N)$
Population

Figure 1:

- However, in practice we have only one sample.

Mimic Resampling

- We propose to mimic this process
  - by drawing B samples $\mathcal{S}_1^\star, \ldots, \mathcal{S}_B^\star$ of size n independently from a population $\mathcal{P}^\star$.

- Ideally, $\mathcal{P}^\star$ will be the study population $\mathcal{P}$,
  - but our sample estimate of $\mathcal{P}$ is the sample $\mathcal{S}$, so we take $\mathcal{P}^\star = \mathcal{S}$ or $\widehat{\mathcal{P}} = \mathcal{S}$.
  - However, we could use any estimate of the study population.

- The sample population has only n units, so without replacement sampling mechanism will immediately exhaust the population. Therefore, we sample **with replacement**.

- Drawing B samples $\mathcal{S}_1^\star, \ldots, \mathcal{S}_B^\star$ of size n from $\mathcal{P}^\star$ with replacement,
  - the sample attribute values $a(\mathcal{S}_1^\star), \ldots, a(\mathcal{S}_B^\star)$ now provide the information on the sampling distribution of interest.

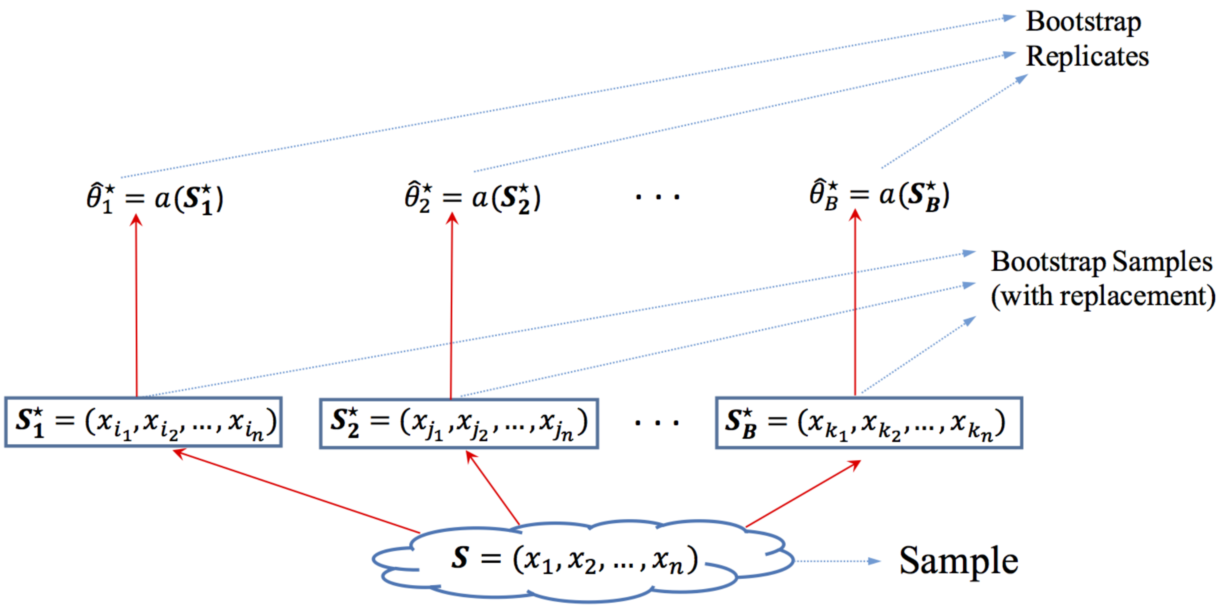

Bootstrap
Replicates
$\hat{\theta}_1^\star = a(\boldsymbol{S}_1^\star)$
$\hat{\theta}_2^\star = a(\boldsymbol{S}_2^\star)$
$\cdots$
$\hat{\theta}_B^\star = a(\boldsymbol{S}_B^\star)$
Bootstrap Samples
(with replacement)
$\boldsymbol{S}_1^\star = (x_{i_1}, x_{i_2}, \ldots, x_{i_n})$
$\boldsymbol{S}_2^\star = (x_{j_1}, x_{j_2}, \ldots, x_{j_n})$
$\cdots$
$\boldsymbol{S}_B^\star = (x_{k_1}, x_{k_2}, \ldots, x_{k_n})$
$\boldsymbol{S} = (x_1, x_2, \ldots, x_n)$
Sample

Figure 2:

The Bootstrap Method

- This approach of mimicking the sampling distribution was named the **bootstrap** method when it was first proposed in 1979 by Bradly Efron.
- The word “bootstrap” conveys the notion of starting something up from nothing as in “pulling oneself over a fence by one’s bootstraps”.
  - It suggests something for nothing, or something impossible to achieve.
- In Efron (1979), other possible names were *Swiss Army Knife*, *Meat Axe*, *Swan-Dive*, *Jack-Rabbit*.
- The Bootstrap path of Inductive inference

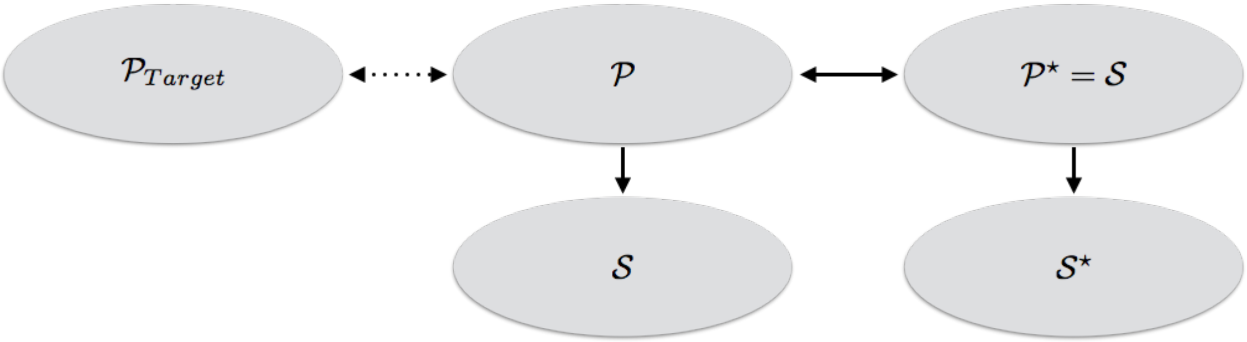


Figure 3:

Why is bootstrapping so important? What does it provide?

Shark Lengths Example

- Suppose we have a sample of $n = 6$ and we are interested in estimating the average shark length.
  - Do we expect the bootstrap to work if n is small?
  - The population has only $N = 65$ units so the sample size is a little under 10 percent of the population.

- We draw one sample $\mathcal{S}$ drawn from $\mathcal{P}$ using simple random sampling without replacement.

```r
set.seed(341)
n <- 6
S <- getSample(1:65, n, replace = FALSE)
S
```

```
## [1] 10 58 24  4 50 46
```

- Then we draw B bootstrap samples from this single sample.
  - There are $n^n = 6^6 = 46656$ possible **bootstrap samples** of size $n = 6$ to select.
  - Here, we choose $B = 10000$ bootstrap samples $S_1^\star, \ldots, S_{10000}^\star$:
  - We might choose a different B depending on the context.

```r
Pstar <- S
B <- 10^4
set.seed(341)
Sstar <- sapply(1:B, FUN =function(b)  getSample(Pstar, n, replace = T))
```

Here we then have a matrix bootstrap samples and the first bootstrap sample contains units

```r
#dim(Sstar)
Sstar[,1]
```

```
## [1] 10 46 24 10 50 50
```

- We then compute whichever attribute might be of interest on each bootstrap sample.
  - e.g. we compute the length average for each bootstrap sample.

```r
avesBootSamp <- sapply(1:B, FUN = function(i) mean(sharks[Sstar[,i], "Length"]))
#length(avesBootSamp)
```

The initial ten bootstrap sample averages are

```r
round(avesBootSamp[1:10],1)
```

```
##  [1] 112.3 128.7 121.8 122.0 121.0 100.5 108.8 110.8 120.3 142.5
```

- The collection bootstrap averages is a population and
    - can be summarized like any other population
    - but is estimate of the sampling distribution of the sample average.

An estimate of the sampling distribution

- The distribution of any attribute over the bootstrap samples $\mathcal{S}_i^\star$ from $\mathcal{P}^\star$ is a **bootstrap estimate** of the distribution of the same attribute over all possible samples $\mathcal{S}_i$ from $\mathcal{P}$

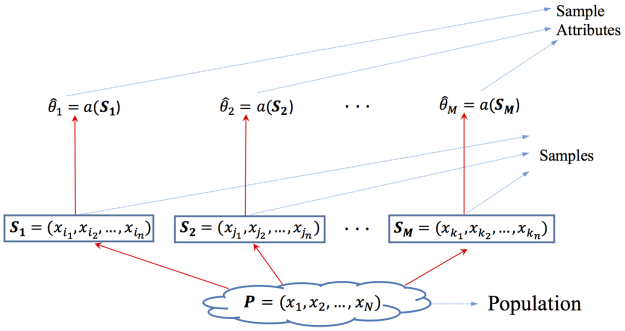

Sample
Attributes
$\hat{\theta}_1 = a(\boldsymbol{S_1})$
$\hat{\theta}_2 = a(\boldsymbol{S_2})$
$\cdots$
$\hat{\theta}_M = a(\boldsymbol{S_M})$
Samples
$\boldsymbol{S_1} = (x_{i_1}, x_{i_2}, \ldots, x_{i_n})$
$\boldsymbol{S_2} = (x_{j_1}, x_{j_2}, \ldots, x_{j_n})$
$\cdots$
$\boldsymbol{S_M} = (x_{k_1}, x_{k_2}, \ldots, x_{k_n})$
$\boldsymbol{P} = (x_1, x_2, \ldots, x_N)$
Population

Figure 4:

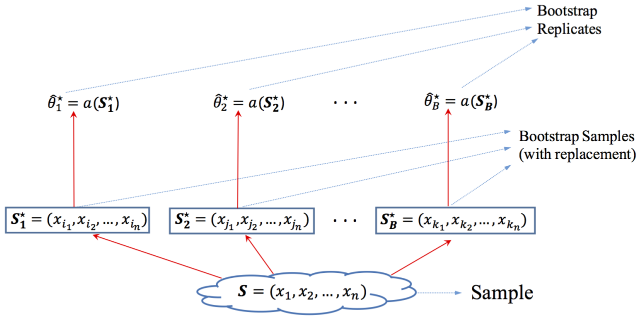

Bootstrap
Replicates
$\hat{\theta}_1^\star = a(\boldsymbol{S_1^\star})$
$\hat{\theta}_2^\star = a(\boldsymbol{S_2^\star})$
$\cdots$
$\hat{\theta}_B^\star = a(\boldsymbol{S_B^\star})$
Bootstrap Samples
(with replacement)
$\boldsymbol{S_1^\star} = (x_{i_1}, x_{i_2}, \ldots, x_{i_n})$
$\boldsymbol{S_2^\star} = (x_{j_1}, x_{j_2}, \ldots, x_{j_n})$
$\cdots$
$\boldsymbol{S_B^\star} = (x_{k_1}, x_{k_2}, \ldots, x_{k_n})$
$\boldsymbol{S} = (x_1, x_2, \ldots, x_n)$
Sample

Figure 5:

Shark Length Comparision

- We can compare the **bootstrap estimate** to resampling from the population. To perform resampling from the population,

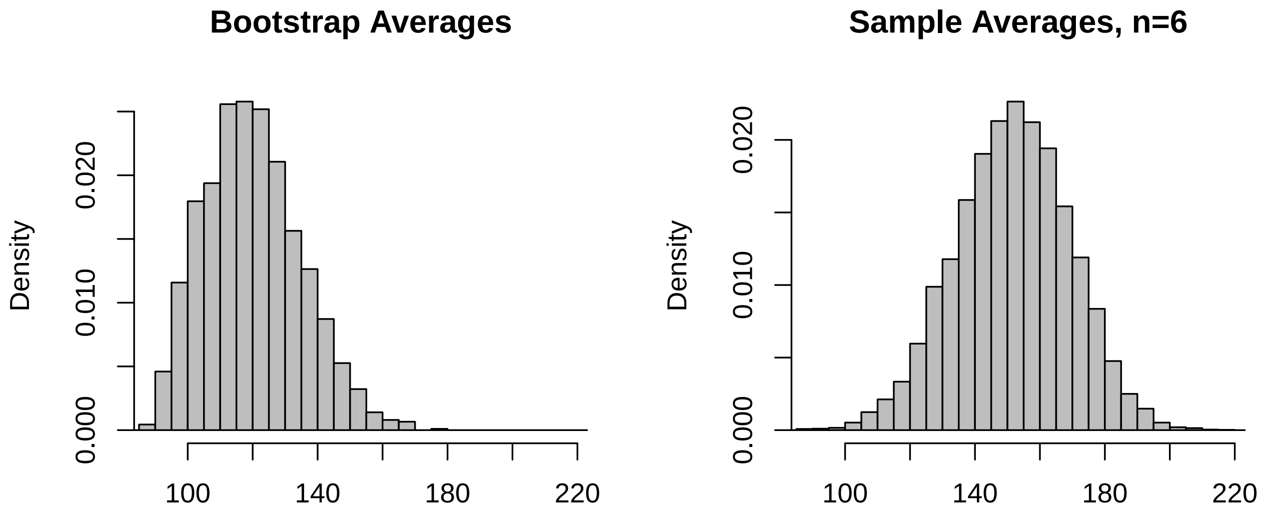


Figure 6: Bootstrap versus Sampling with n=6: averages

- First we have to draw $B = 10,000$ samples from the population and
- calculate the average on each sample.

```
avesSamp <- sapply(1:B, FUN = function(i) mean(sharks[sample(65, n), "Length"]))
#length(avesSamp)
```

The initial ten sample averages are

```
round(avesSamp[1:10],1)
```

```
##  [1] 184.3 160.5 157.2 166.2 178.2 167.0 165.3 138.8 144.3 131.3
```

```
savePar <- par(mfrow=c(1,2))

hPopAve <- hist(extendrange(c(avesSamp,avesBootSamp)), breaks = 50, plot = FALSE)
hist(avesBootSamp, xlim = range(avesSamp), breaks = hPopAve$breaks,
     freq = FALSE, col = "grey", main ="Bootstrap Averages", xlab="")
hist(avesSamp, xlim = range(avesSamp), breaks = hPopAve$breaks,
     freq = FALSE, col = "grey", main ="Sample Averages, n=6", xlab="")
```

- Note that the bootstrap estimate is, at best, as good as $a(\mathcal{S})$ (why?).
- As can been seen, the bootstrap distribution gives a sense of how an attribute varies.
  - e.g., to get some idea of the variability we can take the standard deviation of the bootstrap distribution.

- To easily compare the variability we can construct a histogram of the errors using

$$\begin{aligned}\text{sample error} &= a\left(\mathcal{S}\right) - a\left(\mathcal{P}\right)\\ \text{bootstrap sample error} &= a\left(\mathcal{S}^{\star}\right) - a\left(\mathcal{S}\right)\end{aligned}$$

```
savePar <- par(mfrow=c(1,2))
```

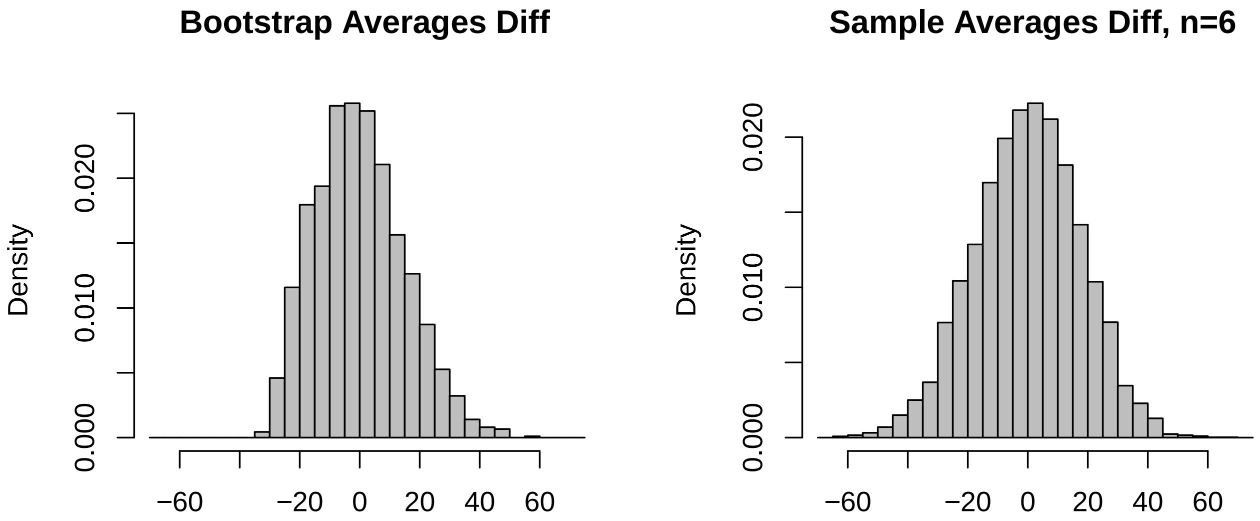


Figure 7: Bootstrap versus Sampling with n=6: averages

```
range.avediff  <- extendrange(
  c(avesSamp- mean(avesSamp),
    avesBootSamp- mean(avesBootSamp)) )

hPopAvediff <- hist(range.avediff,
  breaks = 50, plot = FALSE)

hist(avesBootSamp - mean(avesBootSamp),
     xlim = range.avediff, breaks = hPopAvediff$breaks,
     freq = FALSE, col = "grey",
     main ="Bootstrap Averages Diff", xlab="")


hist(avesSamp - mean(avesSamp),
     xlim = range.avediff, breaks = hPopAvediff$breaks,
     freq = FALSE, col = "grey",
     main ="Sample Averages Diff, n=6", xlab="")
```

Bootstrap Standard Deviation

For any attribute, $a(\mathcal{P})$, the standard deviation of the corresponding sample attribute can be estimated from the bootstrap distributions with

$$\widehat{SD}_\star\left(\widetilde{a}(\mathcal{S}^\star)\right) = \sqrt{\frac{\sum_{b=1}^{B}\left(a(\mathcal{S}_b^\star) - \overline{a}^\star\right)^2}{B-1}}$$

- where $\overline{a}^\star = \sum_{b=1}^{B} a(\mathcal{S}_b^\star)/B$ is the average of the attribute on the bootstrap samples $\mathcal{S}_1^\star, \ldots, \mathcal{S}_B^\star$. Since B is usually large, it does not make any practical difference to use B or $B-1$ in the denominator of the standard deviation.

- This is an estimate of the standard deviation of the sampling distribution for the attribute $a(\mathcal{S})$.

The bootstrap estimate is

```
sdn(avesBootSamp)
```

```
## [1] 14.82945
```

whereas the standard deviation of the sample averages is

```
sdn(avesSamp)
```

```
## [1] 17.61667
```

The Average

- In the case of the arithmetic average $a(\mathcal{S}) = \sum_{u\in\mathcal{S}} y_u/n$
- The bootstrap estimate of the its standard deviation is

$$\widehat{SD}_\star(\overline{Y}) = \sqrt{\frac{\sum_{b=1}^{B}\left(\overline{y}^\star_b - \overline{y}^\star\right)^2}{B-1}}$$

  where $\overline{y}^\star = \frac{\sum_{b=1}^{B}\overline{y}^\star_b}{B}$.
- The standard deviation can also be estimated with

$$\widehat{SD}(\overline{Y}) = \frac{\widehat{\sigma}}{\sqrt{n}}\sqrt{\frac{N-n}{N-1}} \qquad \text{where} \qquad \widehat{\sigma} = \sqrt{\frac{\sum_{u\in\mathcal{S}}\left(y_u - \overline{y}\right)^2}{n}}.$$

- Comparing these two estimates on the the length variable in shark encounters data, yields

```
n=6
N=dim(sharks)[1]
c(sdn(avesBootSamp), sdn(sharks[S, "Length"])/sqrt(n)*sqrt((N-n)/(N-1)) )
```

```
## [1] 14.82945 14.31729
```

If the sample is not a good representative of the population, these two numbers might be quite different.

Comparing two estimators of the standard deviation

- We can compare the two estimators of standard deviation by a generating $m = 2000$ samples of size $n = 6$ and for each sample:
  - calculate $\frac{\widehat{\sigma}}{\sqrt{n}}\sqrt{\frac{N-n}{N-1}}$ and then
  - generate $B = 200$ bootstrap samples then to obtain the bootstrap standard deviation.

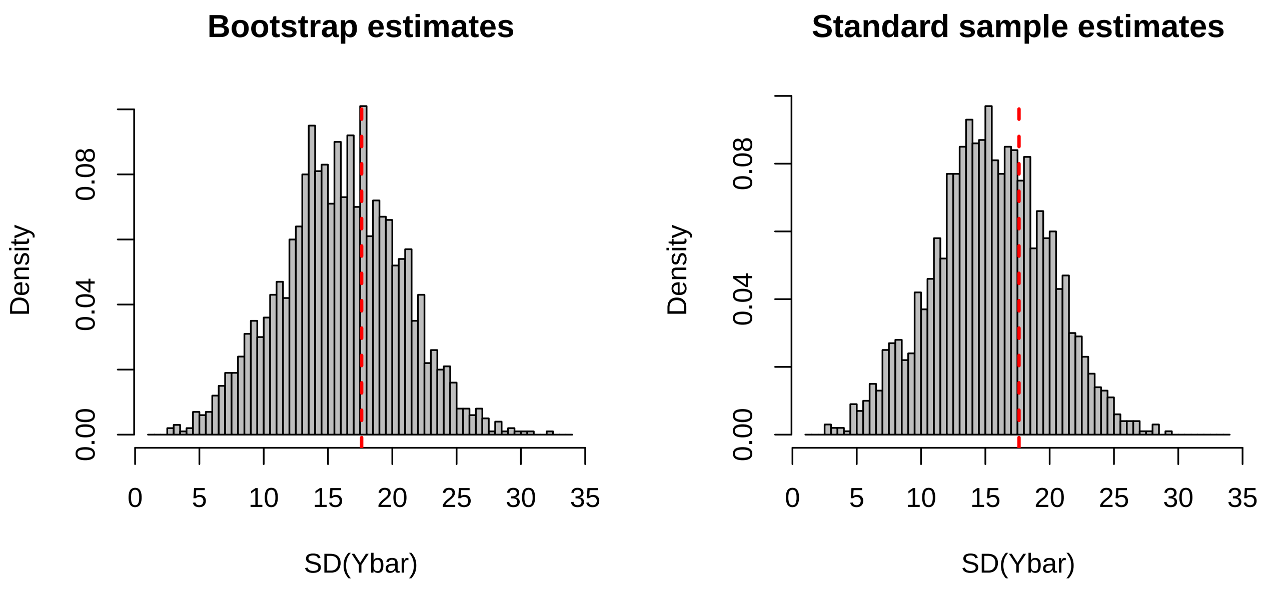


Figure 8: Comparing estimators of the standard deviation of the average

```
set.seed(341)
numSamps <-2000
n <- 6

sampSes <- sapply(1:numSamps, FUN = function(i) getSample(popSharks, n))
B <- 200

valsBoot = t(apply(sampSes,2, function(sam=NULL, B=NULL) {
  avesTemp = unlist(Map( function(i) {
      mean(sharks[sample(sam, n, replace = TRUE), "Length"]) }, 1:B) )

  c( sdn(avesTemp), sdn( sharks[sam, "Length"] )/sqrt(n)*sqrt((N-n)/(N-1)))
  }, B=B ))



colnames(valsBoot) <- c("sdBoot", "sdStandard")

par(mfrow=c(1,2))
hTmp <- hist(extendrange(valsBoot), breaks = 50, plot = FALSE)
hist(valsBoot[,"sdBoot"], xlim = extendrange(valsBoot),
     breaks = hTmp$breaks, freq = FALSE, col = "grey",
     main ="Bootstrap estimates", xlab="SD(Ybar)")
abline(v=sdn(avesSamp), col="red", lty=2, lwd=2)

hist(valsBoot[,"sdStandard"], xlim =extendrange(valsBoot),
     breaks = hTmp$breaks, freq = FALSE, col = "grey",
     main ="Standard sample estimates", xlab="SD(Ybar)")
abline(v=sdn(avesSamp), col="red", lty=2, lwd=2)
```

- The two histograms are not too dissimilar in shape.
- We can further explore the differences between these two estimators.

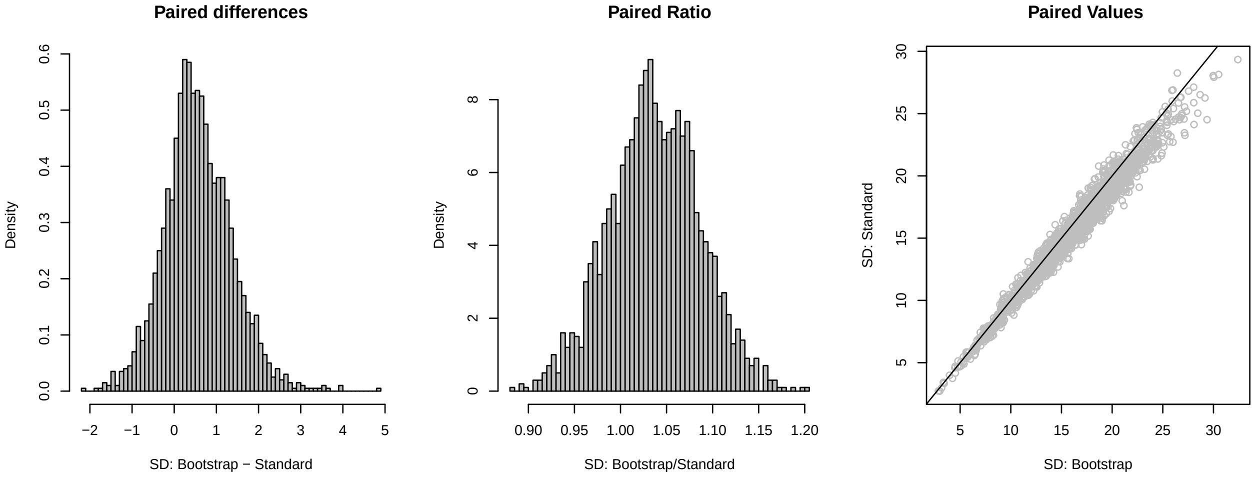


Figure 9: Paired differences in SD(Ybar) esimates. The bootstrap estimator of standard deviation has produced, on average, slightly larger values than the standard.

- We can compare these two populations using the usual methods
  - Note: we have paired data because we calculated the two estimates on each sample.

```
par(mfrow=c(1,3))
hist(valsBoot[,"sdBoot"] - valsBoot[,"sdStandard"], main = "Paired differences", breaks=50, col="grey",

hist(valsBoot[,"sdBoot"] / valsBoot[,"sdStandard"], main = "Paired Ratio", breaks=50, col="grey", freq=

plot(valsBoot[,"sdBoot"], valsBoot[,"sdStandard"], main = "Paired Values ",col="grey", xlab="SD: Bootst
abline(a=0,b=1)
```

- The third plot confirms that the bootstrap method may slightly over-estimate, the population SD.

Aside: n **Versus** $n - 1$

- In the above interval calculations the function `sdn(...)` was introduced and used which is an implementation of

$$\widehat{\sigma} = \sqrt{\frac{\sum_{u \in \mathcal{S}} \left(y_u - \overline{y}\right)^2}{n}}$$

  which has n as a divisor. The built-in function `sd(...)` in R is an implementation of

$$\widehat{\sigma} = \sqrt{\frac{\sum_{u \in \mathcal{S}} \left(y_u - \overline{y}\right)^2}{n - 1}}$$

- For bootstrap interval calculations, a divisor of n is preferred.
  - This estimate has the advantage of being **replication invariant**.

  - Replication invariant estimates are preferred and are often called **plug in estimates** in the bootstrap literature e.g. see Efron and Tibshirani (1994) .

- When n is reasonably large, there will be little practical difference between the two.

Estimating the sampling bias

- Recall that the **sampling bias**

$$\text{Sampling Bias} = E[a(\mathcal{S})] - a(\mathcal{P})$$

- We can use the bootstrap to estimate sampling bias via

$$\widehat{\text{sample bias}} = \text{average bootstrap sample error} = \overline{a}^{\star} - a\left(\mathcal{S}\right)$$

  - where $\overline{a}^{\star} = \sum_{b=1}^{B} a(\mathcal{S}_b^{\star})/B$ is the average of the attribute on the bootstrap samples $\mathcal{S}_1^{\star}, \ldots, \mathcal{S}_B^{\star}$.

Biased Corrected Estimators

- Some estimators are unbiased and some are biased.
  - We would like to "correct"" biased estimators.
  - i.e. make a biased estimator unbiased.

- If $a(\mathcal{S})$ was biased and we knew the bias then
  - we could add the correction to our attribute or estimator to make a new attribute $a^{\star}(\mathcal{S})$ that is unbiased.

$$a^{\star}(\mathcal{S}) = a(\mathcal{S}) - \text{Sampling Bias}$$

    because

$$E[a^{\star}(\mathcal{S})] = E[a(\mathcal{S}) - \text{Sampling Bias}] = E[a(\mathcal{S}) - E[a(\mathcal{S})] + a(\mathcal{P})] = a(\mathcal{P})$$

- Then the Bootstrap biased correct estimate is

$$a(\mathcal{S}) - \widehat{\text{bias}}\left[a(\mathcal{S})\right] = a\left(\mathcal{S}\right) - \left[\overline{a}^{\star} - a\left(\mathcal{S}\right)\right] = 2a\left(\mathcal{S}\right) - \overline{a}^{\star}$$

Example with different Attributes

- Using the sample, 10, 58, 24, 4, 50, 46, we can estimate their bias and standard deviation of the corresponding estimators for the shark lengths:
  - median
  - standard deviation
  - skewness

The required functions and the samples estimates are

```
skew  <- function(z) { 3*(mean(z) - median(z))/sdn(z) }

sam.len = sharks[S, "Length"]
sam.est = c(median(sam.len), sdn(sam.len), skew(sam.len))
round(sam.est,2)
```

```
## [1] 105.00  36.53   1.25
```

- Generating the bootstrap samples and calculating the attributes on each bootstrap sample

```
bootAttr <- apply(Sstar, 2, function(sam, len) {
                pop = len[sam]
                c( median(pop), sdn(pop), skew(pop) )
                },
                len = sharks[, "Length"] )
# dim(bootAttr)
# the rows of boot Attr are the attributes (median,std. dev., and skewness)
## from 10,000 bootstrap samples.
```

The bootstrap estimate of the sampling distribution's standard deviation is

```
round(apply(bootAttr,1,sd),3)
```

```
## [1] 18.370 11.643  0.825
```

The bootstrap estimates of their sampling bias is

```
bias.est = apply(bootAttr,1,mean) - sam.est
round(bias.est,2)
```

```
## [1]  6.5 -5.3 -0.5
```

The biased corrected bootstrap estimates are:

```
round( sam.est - bias.est, 2)
```

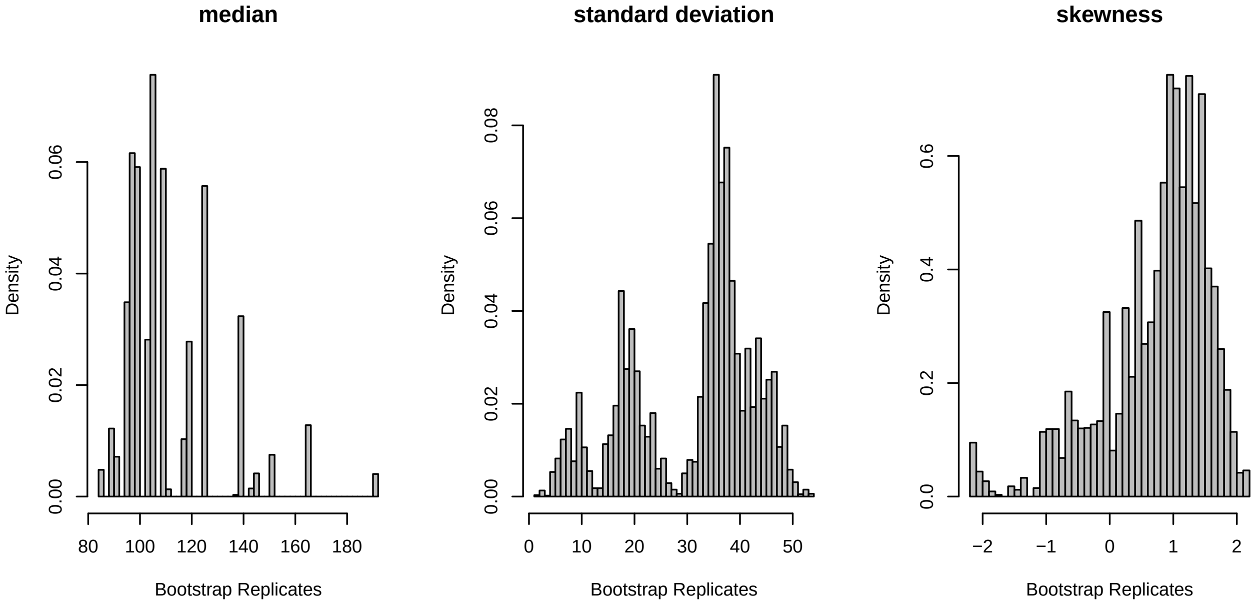
```
## [1] 98.50 41.82  1.74
```

```
###
###
### or equivalently,
### round( 2*sam.est - apply(bootAttr,1,mean)  ,2)
```

Estimates of their sampling distributions (no correction to the bias applied):

```
par(mfrow=c(1,3))
namHist = c("median", "standard deviation", "skewness")

for (i in 1:3) hist(bootAttr[i, ], main = namHist[i], xlab="Bootstrap Replicates",
     breaks=50, col="grey", freq=FALSE)
```



4.5.2 Bootstrap confidence intervals

- The bootstrap distribution provides a proxy for the sampling distribution for any sample attribute $a(\mathcal{S})$.
  - So we can use it construct (at least approximate) confidence intervals for the unknown population attribute $a(\mathcal{P})$.

- If the bootstrap distribution is approximately normal then
  - noting that confidence intervals for sample averages, for example, have the following structure

$$\left[\overline{y} - c\widehat{SD}(\overline{Y}), \quad \overline{y} + c\widehat{SD}(\overline{Y})\right]$$

- Picking c such that $P(Z \leq c) = 1 - \alpha/2$ generates a $100(1-\alpha)\%$ confidence interval (under Gaussian assumption).

- Rather than using the $\widehat{\sigma}/\sqrt{n} \times \sqrt{(N-n)/(N-1)}$ for $\widehat{SD}(\overline{Y})$,
  - we might use standard deviation of the bootstrap distribution of $\overline{Y}$ to produce the standard deviation estimate $\widehat{SD}(\overline{Y})$.
  - The attraction of this approach, if it works, is that the same approach could be used for any attribute $a(\mathcal{S})$.

- A 95% **standard bootstrap interval** is

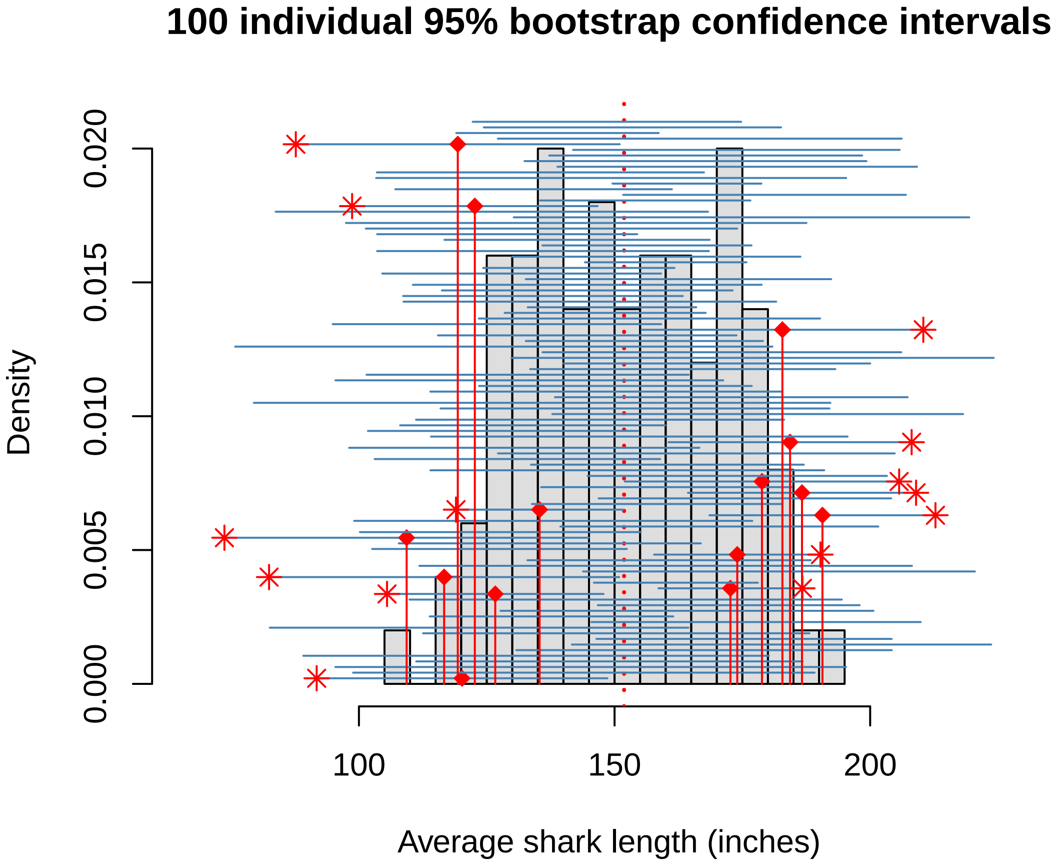
$$a(\mathcal{S}) \pm 1.96\widehat{SD}_{boot}\left(a(\mathcal{S})\right)$$

  where $\widehat{SD}_{boot}$ is the bootstrap estimate of the standard deviation. What does 95% mean here?

- If we use the standard bootstrap interval then we should investigate its properties such as determining if we obtain the proper coverage probability.

Estimating the standard bootstrap interval coverage probability

- The average shark length of great whites was 151.86 inches (65 encounters).
- Suppose we are interested in a 95% confidence interval for the population mean.
- To get an estimate of the coverage probability for the bootstrap confidence interval, we generate 1000 samples of size $n = 6$ and for each sample
  - generate B bootstrap samples to obtain the bootstrap estimate of the standard deviation, $\widehat{SD}(\overline{Y})$, and
  - then construct the interval with $c = 1.96$.



- As can be seen, only 86 of the 100 bootstrap intervals actually cover the average in the population.
  - This is a much lower coverage proportion than the expected 95 % suggested by using $c = 1.96$.

- Under-coverage is perhaps not unexpected as the c value is based on a Gaussian model which may not apply.

4.5.2.2 Percentile method

- The bootstrap distribution provides a proxy for the sampling distribution for any sample attribute $a(\mathcal{S})$.
  - So we can use it construct (at least approximate) confidence intervals for the unknown population attribute $a(\mathcal{P})$.

- If the bootstrap distribution is approximately a Gaussian distribution then confidence intervals inspired by the familiar structure
$$a(\mathcal{S}) \pm c \times \widehat{SD}(a(\mathcal{S}))$$
  - However, what if the bootstrap distribution is not approximately Gaussian?

- Then why not simply use the bootstrap distribution directly to construct a confidence interval?

Bootstrap versus Sampling Distribution

- Recall we compared
  - the Bootstrap distribution (based on sampling with replacement with $n = 6$ from a single sample) versus
  - Sampling Distribution (based on sampling without replacement with $n = 6$ from the population)
- Using the quantiles from sampling distribution we can obtain an interval that covers 95% of the sample averages.
  - This is (116.33, 184.83)

- Using the quantiles from bootstrap distribution we can obtain an estimate of the interval that covers 95% of the bootstrap averages.
  - This is (94.83, 151.83)

- The population average is 151.86

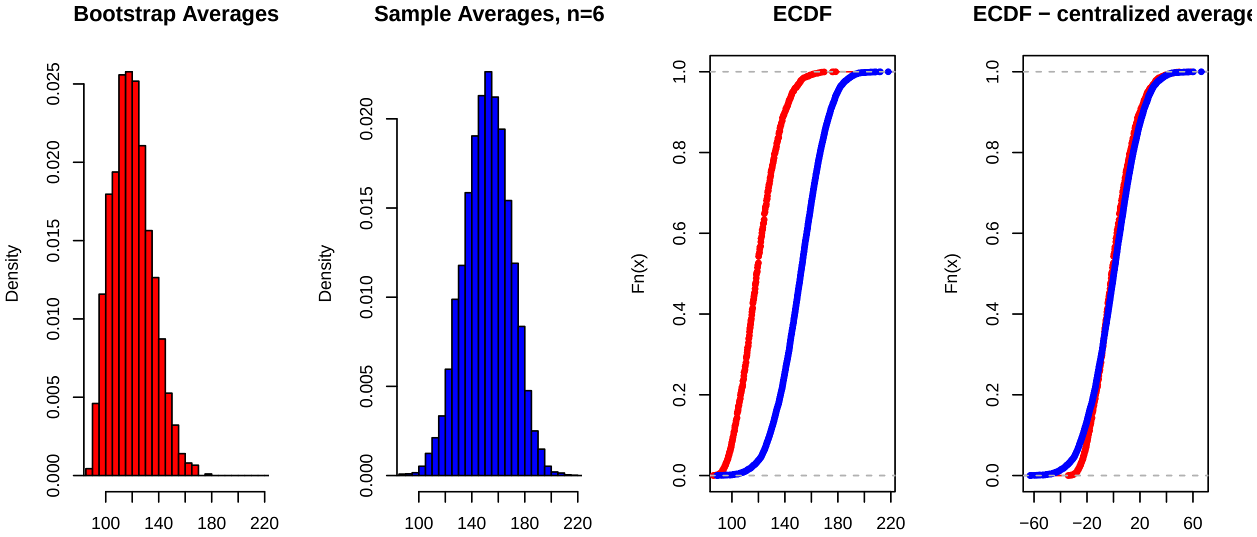


Figure 10: Bootstrap versus Sampling with n=6: averages

Percentile Method

- For a given sample $\mathcal{S}$ generate B bootstrap samples $\mathcal{S}_1^\star, \ldots, \mathcal{S}_B^\star$
  - by sampling with replacement from the sample $\mathcal{S}$.
  - For the b^{th} bootstrap sample $(b = 1, ..., B)$, calculate $a_b = a(\mathcal{S}_b^\star)$.

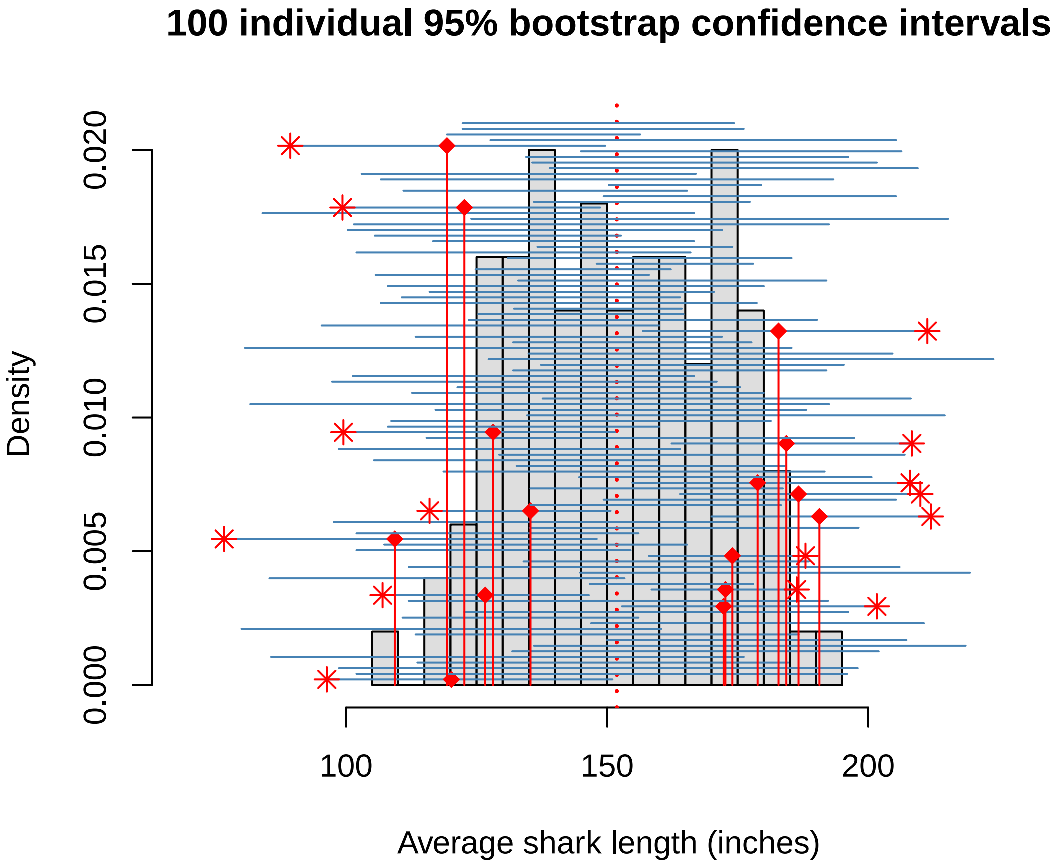
- From the values $a_1, \ldots, a_B$, find
  - $a_{lower} = Q_a\,(p/2)$ and $a_{upper} = Q_a\,(1 - p/2)$
  - Then a $100(1-p)\%$ confidence interval using the percentile method is $[a_{lower}, a_{upper}]$.

- This approach is **equivariant** to any one to one transformation (increasing function) of the attribute, say $T(a(\mathcal{P}))$.
  - i.e., the corresponding interval for $T(a(\mathcal{P}))$ is simply $[T(a_{lower}), T(a_{upper})]$!
  - So, we only need to determine the values a_{lower} and a_{upper} once for $a(\mathcal{P})$ and we have them for any $T(a(\mathcal{P}))$.
  - If the one to one transformation $T(a(\mathcal{P}))$ is monotonically decreasing, then the interval is $[T(a_{upper}), T(a_{lower})]$.

Assessing the coverage probability

- To estimate the percentile bootstrap interval coverage probability using a 95% confidence level.
- We generate 1000 samples of size $n = 6$ and for each sample $a_b = a(\mathcal{S}_b^\star)$.
  - From the values $a_1, \ldots, a_B$, find

- $a_{lower} = Q_a\,(p/2)$ and $a_{upper} = Q_a\,(1-p/2)$
- Report the 95% percentile confidence intervals as $[a_{lower}, a_{upper}]$.



- As can be seen, 85 of the 100 bootstrap intervals actually cover the average in the population.
  - This is a much lower coverage proportion than the expected 95 % suggested by using $c = 1.96$.

Comments

- Simplicity is the attraction of this method, and explains its continued popularity.
- This method is transformation invariant
- The coverage probability is often incorrect if the distribution of estimator is not nearly symmetric (Efron and Tibshirani, p. 178).
  - To improve the coverage probability we have to consider pivotal quantities or approximate quantities.

4.5.2.1 Bootstrap-t confidence intervals

- Some sampling experiments showed that

  $$Z = \frac{\widetilde{a}(\mathcal{S}) - a(\mathcal{P})}{\widetilde{SD}(\widetilde{a}(\mathcal{S}))}$$

  was approximately pivotal for $a(\mathcal{S}) = \overline{y}$

  - its histogram (over all possible samples) was well approximated by a t-density.
  - This suggests instead of using a c from a Gaussian, we use a t-distribution on $n-1$ degrees of freedom.

- Better still instead of approximate distribution (a t-distribution on $n-1$ degrees of freedom) we use its actual distribution to determine the c value.
  - If $a(\mathcal{P})$ is the median or a measure of skewness, we would not expect the t-distribution to be a good approximation.

- We have seen that the bootstrap is a method to approximate
  - the average error,
  - the standard deviation,
  - sampling distribution.

- Here we will use the bootstrap to estimate the sampling distribution of

  $$Z = \frac{\widetilde{a}(\mathcal{S}) - a(\mathcal{P})}{\widetilde{SD}(\widetilde{a}(\mathcal{S}))}$$

  and use it to construct confidence intervals.

Sampling Distribution Approximation

- The t-distribution can approximate the sampling distribution of certain sample attributes.
  - This requires $\widetilde{a}(\mathcal{S})$ to be approximately Gaussian over all possible samples and
  - an estimate of the standard deviation.

- The bootstrap distribution can approximate the sampling distribution of a sample attribute.
  - This requires computation.
  - However the bootstrap automatically adjusts its shape (and hence quantiles, etc.) to the form of $\widetilde{a}(\mathcal{S})$.

- When we sample (say, $n = 6$) from all possible samples, we use the t-distribution to approximate

$$Z = \frac{\widetilde{a}(\mathcal{S}) - a(\mathcal{P})}{\widetilde{SD}(\widetilde{a}(\mathcal{S}))}$$

```
ZPop <- (avesSamp - mean(sharks[,"Length"]))/sdn(avesSamp)
```

- To use the bootstrap to approximate this quantity replace
  - the sample $\mathcal{S}$ with the bootstrap sample $\mathcal{S}^\star$ and
  - the population $\mathcal{P}$ with an estimate $\mathcal{P}^\star$, (the sample $\mathcal{S}$) and

$$Z^\star = \frac{\widetilde{a}(\mathcal{S}^\star) - a(\mathcal{P}^\star)}{\widetilde{SD}(\widetilde{a}(\mathcal{S}^\star))} = \frac{\widetilde{a}(\mathcal{S}^\star) - a(\mathcal{S})}{\widetilde{SD}(\widetilde{a}(\mathcal{S}^\star))}$$

```
ZBoot <- (avesBootSamp - mean(sharks[Pstar,"Length"]))/sdn(avesBootSamp)
```

The bootstrap-t

- For a given sample $\mathcal{S}$
  - calculate $a(\mathcal{S})$ and $\widehat{SD}\left(a(\mathcal{S})\right)$.

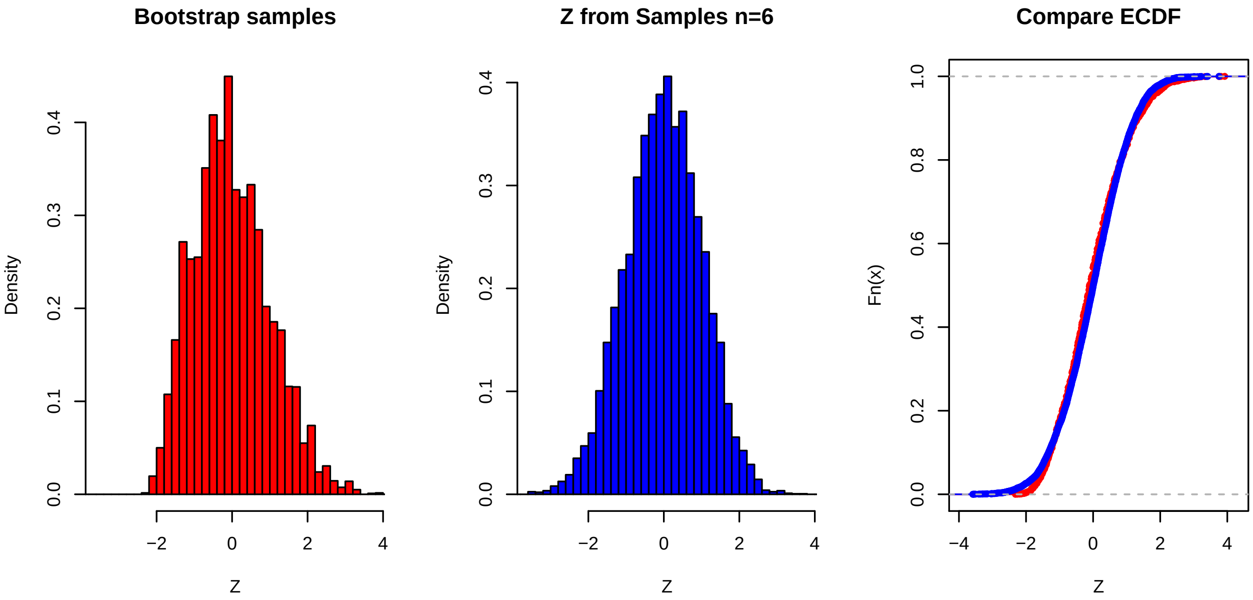
- Generate B bootstrap samples $\mathcal{S}_1^\star, \ldots, \mathcal{S}_B^\star$ from $\mathcal{S}$ and for each bootstrap sample:
  - Calculate $a(\mathcal{S}_b^\star)$, $\widehat{SD}\left(a(\mathcal{S}_b^\star)\right)$ and
$$z_b = \frac{a(\mathcal{S}_b^\star) - a(\mathcal{S})}{\widehat{SD}\left(a(\mathcal{S}_b^\star)\right)}.$$

- From the values $z_1, \ldots, z_B$, find
  - $c_{lower} = Q_z\left(p/2\right)$ and $c_{upper} = Q_z\left(1 - p/2\right)$

- Then a $100(1-p)\%$ bootstrap-t confidence interval is
$$\left[a(\mathcal{S}) - c_{upper} \times \widehat{SD}\left(a(\mathcal{S})\right), a(\mathcal{S}) - c_{lower} \times \widehat{SD}\left(a(\mathcal{S})\right)\right].$$
  - **Note** the signs and positions of the constants c_{lower} and c_{upper} in the interval definition.

- This method (so far) requires an analytic form to calculate $\widehat{SD}\left(a(\mathcal{S}_b^\star)\right)$, which may or may not be available.

Sampling Distribution Comparison

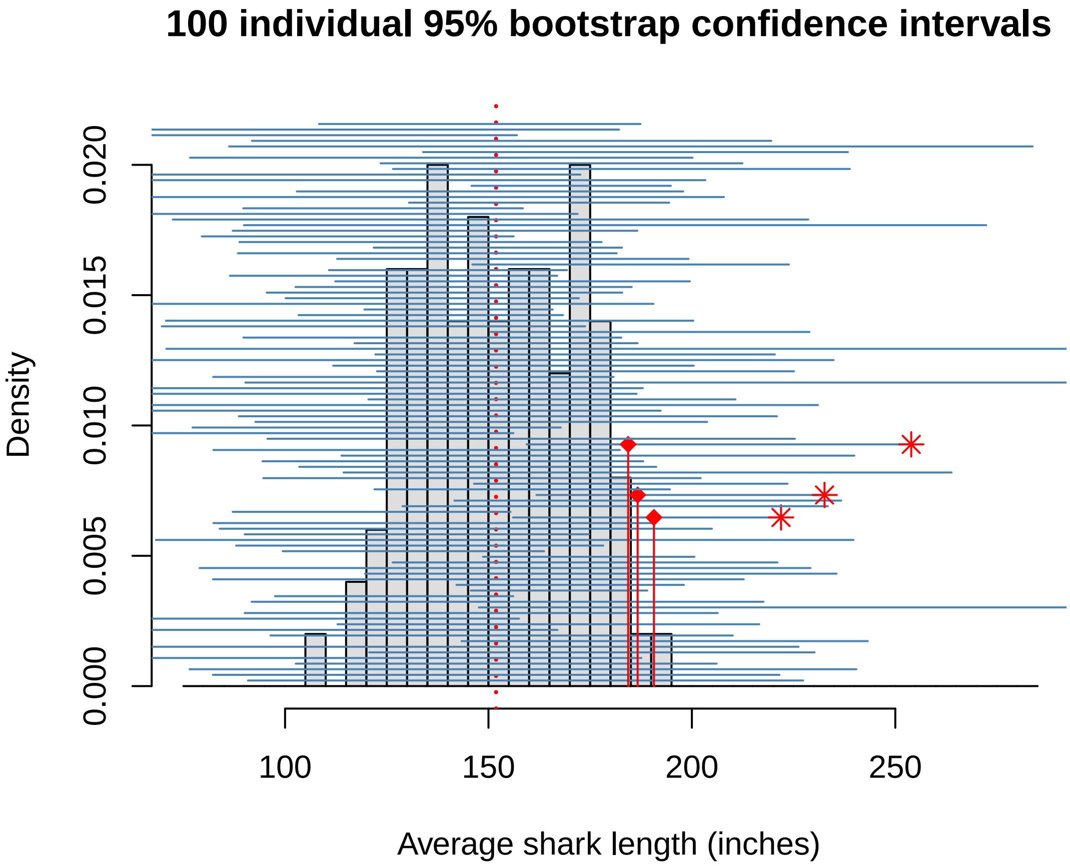


We use the bootstrap distribution of T to find values c_{lower} and c_{upper} such that

$$Pr(c_{lower} \leq Z \leq c_{upper}) = (1-p)$$

with $(1-p)$ being the intended coverage probability.

Bootstrap-t Coverage probability



- Now 97 of the 100 bootstrap intervals cover the average in the population.
  - These intervals will be wider due to the bootstrap estimates of c_{lower} and c_{upper} than earlier intervals.

A thought on Standard Deviations

- When $a(\mathcal{S}) = \overline{y}$,
  - we have an analytic form for its standard deviation, namely

$$SD(\overline{Y}) = \frac{\sigma}{\sqrt{n}} \times \sqrt{\frac{N-n}{N-1}}$$

  - Replacing σ by $\widehat{\sigma}$ gives an estimate $\widehat{SD}(\overline{Y})$ based on the sample values y_u for $u \in \mathcal{S}$.

- Often, an analytic solution is not available for $\widehat{SD}(a(\mathcal{S}))$
  - An estimate can be obtained by using the bootstrap,
  - The standard deviation of the bootstrap values $a(\mathcal{S}_1^\star), \ldots, a(\mathcal{S}_B^\star)$.

- However, in the bootstrap-t, we need an estimate of $SD(a(\mathcal{S}_b^\star))$ for each bootstrap sample $\mathcal{S}_b^\star$.
  - As an analytic form may not be available for $a(\mathcal{S})$, how can we get an estimate?

A Double Double

- **The Double Bootstrap** is general solution which applies the bootstrap method to the bootstrap sample $\mathcal{S}_b^\star$.

- To apply a **bootstrap within a bootstrap**, for each bootstrap sample $\mathcal{S}_b^\star$
  - we calculate a bootstrap estimate of the standard deviation by generating D bootstrap samples, $\mathcal{S}_1^{\star\star}, \ldots, \mathcal{S}_D^{\star\star}$, from $\mathcal{S}_b^\star$.

- Just as with the original bootstrap, this additional bootstrap adds another link in the chain of the inductive path as shown below.

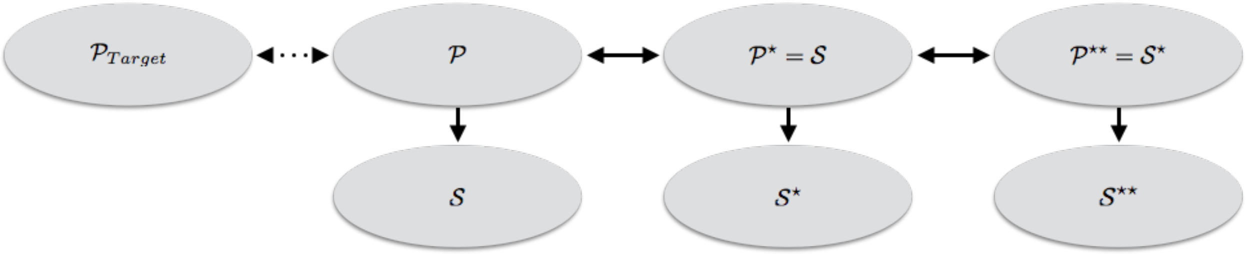

$\mathcal{P}_{Target}$
$\mathcal{P}$
$\mathcal{P}^\star = \mathcal{S}$
$\mathcal{P}^{\star\star} = \mathcal{S}^\star$
$\mathcal{S}$
$\mathcal{S}^\star$
$\mathcal{S}^{\star\star}$

Figure 11:

Example

- The `bootstrap_t_interval` function (see below), requires an attribute function `a` which will calculate $a(\mathcal{S})$ for any $\mathcal{S} \subset \mathcal{P}$ as an argument.

```
bootstrap_t_interval <- function(S, a,
                                  confidence = 0.95,
                                  B = 1000, D = 30){
  ## Here S is the sample, a is a scalar-valued function a(S) of a sample S
  ## which returns the value for S of that attribute of interest
  ## confidence is the level of confidence
  ## B is the outer bootstrap count of replicates used to
  ## calculate the lower and upper limits
  ## D the inner bootstrap count of replicates used to
  ## estimate the standard deviation of the sample attribute
  ## for each (outer) bootstrap sample
  Pstar <- S
  aPstar <- a(Pstar)
  ## get (outer) bootstrap values
  bVals <- sapply(
    1:B,
    FUN = function(b) {
      Sstar <- getSample(Pstar, sampleSize, replace = TRUE)
      aSstar <- a(Sstar)
      ## get (inner) bootstrap values to
      ## estimate the SD
      Pstarstar <- Sstar
      SD_aSstar <- sdn(
        sapply(1:D,
               FUN = function(d){
                 Sstarstar <- getSample(Pstarstar, sampleSize, replace = TRUE)
                 ## return the attribute value
                 a(Sstarstar)
               }
        )
      )
      z <- (aSstar - aPstar)/SD_aSstar
      ## Return the two values
      c(aSstar = aSstar, z = z)
    })
  SDhat <- sdn(bVals["aSstar",])
  zVals <- bVals["z",]
  ## Now use these zVals to get the lower and upper
  ## c values.
  cValues <- quantile(zVals,
                      probs = c((1 - confidence)/2,  (confidence +1)/2),
                      na.rm = TRUE)
  cLower <- min(cValues)
  cUpper <- max(cValues)
  interval <- c(lower = aPstar - cUpper * SDhat,
                middle = aPstar,
                upper = aPstar - cLower  * SDhat)
  interval
}
```

- In the case of the average shark length, this can be written as

```r
a <- function(S) {mean(sharks[S, "Length"])}
```

A single bootstrap-*t* 95% confidence interval is

```r
S <- getSample(popSharks, size = 5, replace = TRUE)

bootstrap_t_interval(S, a)
```

```
##     lower    middle     upper
##  69.64893 139.40000 201.72852
```

Assessing the coverage probability

- As before, we can conduct an experiment to assess the coverage probability for these intervals
    - This experiment takes a fair time to complete

```r
confidence <- 0.95
B <- 1000
set.seed(341)

intervals <- sapply(1:numIntervals,
    FUN = function (i) {
    bootstrap_t_interval(samps[,i], a = a,
          confidence = confidence, B=B, D=30) } )
```

- Exactly 96 of the 100 bootstrap intervals cover the population attribute.

Comments

- Using the bootstrap-*t* intervals perform better than the percentile and the standard bootstrap intervals (based on the Gaussian distribution).

- According to [Efron and Tibshirani (1994), pp. 161-162] the bootstrap-*t* intervals are best suited for attributes which measure "location parameters" like the average, the median, a particular quantile, et cetera.

- For attributes that do not measure location, it may be necessary to first transform the attribute to a scale that produces bootstrap-*t* that have good coverage probabilities,
    - Or consider using the percentile method.

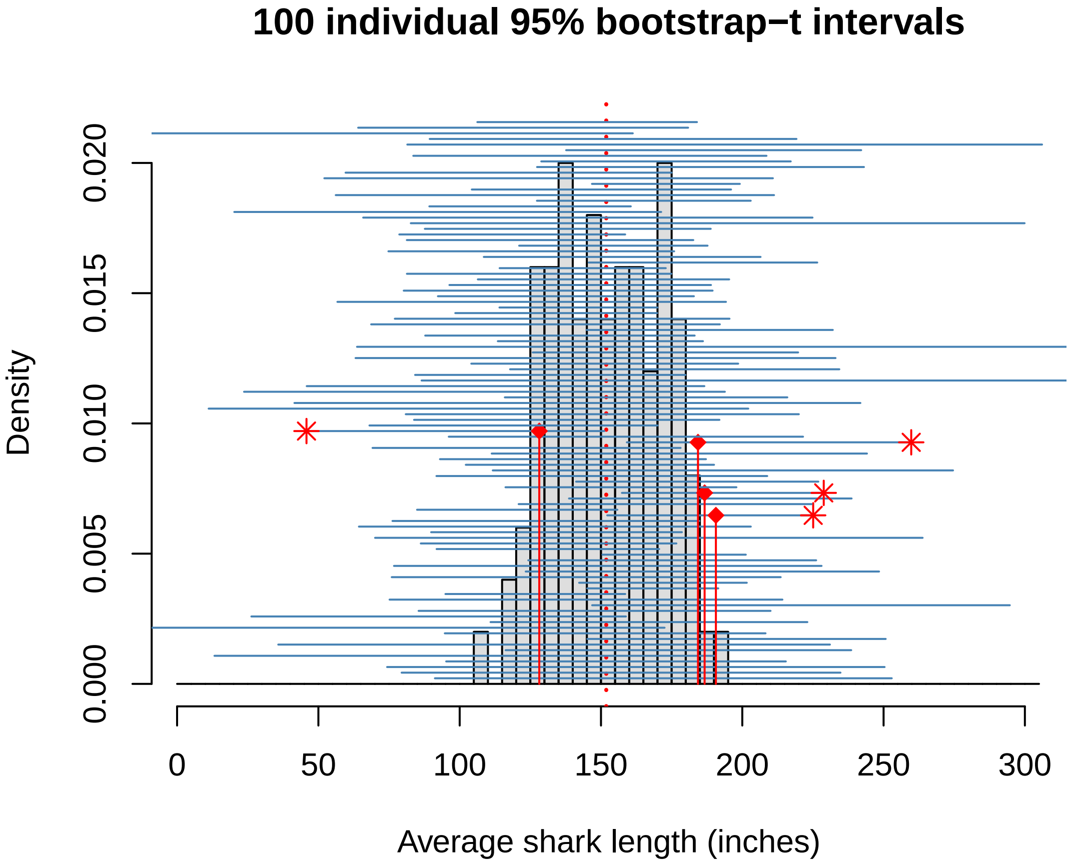


Figure 12: General bootstrap-t intervals