Nonparametric Bootstrap

Elvan Ceyhan

11/21/2022

```
library(nptest)
```

1 Introduction

1.1 Motivation and Goals

Nonparametric bootstrap sampling offers a robust alternative to classic (parametric) methods for statistical inference. Unlike classic statistical inference methods, which depend on parametric assumptions and/or large sample approximations for valid inference, the nonparametric bootstrap uses computationally intensive methods to provide valid inferential results under a wide collection of data generating conditions.

The default install of `R` comes with the `boot` package, which is a collection of bootstrap functions that were originally designed for `S` (the predecessor of `R`). The `bootstrap` package, which is written by the creator of the bootstrap (Bradley Efron), is another popular package for nonparametric bootstrapping. Both the `boot` and `bootstrap` packages have several good features as well as some undesirable characteristics. We will use the `np.boot` function in the `nptest` `R` package, which combines the strengths of the bootstrap functionalities in the `boot` and `bootstrap` packages.

1.2 Sampling with Replacement in `R`

The nonparametric bootstrap involves randomly sampling data with replacement to form a "new" sample of data, which is referred to as a *bootstrap sample*.

Given a (data) vector $\mathbf{x} = (x_1, \ldots, x_n)^t$ of length (or size) n, sampling with replacement involves forming a new vector

$$\mathbf{x}^* = (x_1^*, \ldots, x_n^*)^t$$

where each x_i^* is independently sampled from $\mathbf{x}$ with equal probability given to each observation, i.e., $P(x_i^* = x_j) = 1/n$ for all i, j.

In `R`, sampling with replacement can be conducted using the `sample()` function with the `replace = TRUE` argument.

```
#generate data
x <- letters[1:5]
x
```

```
## [1] "a" "b" "c" "d" "e"
```

```r
#set random seed (for reproducibility)
set.seed(1)

#without replacement (default)
sample(x)
```

```
## [1] "a" "d" "c" "e" "b"
```

```r
# with replacement
sample(x, replace = TRUE)
```

```
## [1] "e" "c" "b" "c" "c"
```

Note that sampling without replacement is equivalent to permuting the original observations, so each original observation appears once in the resampled vector. In contrast, sampling with replacement produces a new vector that could contain anywhere from 0 to n occurrences of each original observation.

1.3 Parametric Statistics Primer

Parameters and Statistics

Inferential statistical methods involve specifying some population of interest, and using a sample of data to infer things about the population. Let X denote the random variable of interest, which is assumed to have some distribution function $F(x) = P(X < x)$. Suppose that the distribution function depends on the parameter $\theta = T(F)$, where $T()$ is some function of F. From the frequentist perspective, the parameter θ is assumed to be some *unknown constant* that describes the probabilistic nature of the population.

Suppose we have a random sample of data $\mathbf{x} = (x_1, \ldots, x_n)^t$ from the population F, where each x_i is an independent and identically distributed (iid) realization of the random variable X. Any function of the sample of data is a *statistic*, which is itself a random variable. For example, suppose that we calculate some estimate of the parameter such as $\hat{\theta} = s(\mathbf{x})$, where $s()$ is some function of $\mathbf{x}$. The estimate $\hat{\theta}$ is a random variable, since it is a function of the random sample $\mathbf{X}$.

Sampling Distributions

The statistic $\hat{\theta} = s(\mathbf{X})$ is a random variable that has some probability distribution, which will be denoted by $G(\theta) = P(\hat{\theta} < \theta)$. The distribution G is referred to as the *sampling distribution* of the statistic $\hat{\theta}$, given that it describes the probabilistic nature of the statistic $\hat{\theta}$ given a random sample of n observations from the population distribution F.

The form of the sampling distribution G will depend on three things:

1. the original data generating distribution F,
2. the function $s()$ used to calculate the statistic,
3. the sample size n.

For certain combinations of F, s, and n, the sampling distribution will be known exactly. However, in many situations, the sampling distribution of $\hat{\theta}$ is only known asymptotically, i.e., as $n \to \infty$. And, for non-standard statistics, the sampling distribution may not even be known asymptotically.

Example 1: Mean

Suppose that the parameter is the population mean $\theta = E[X]$, and the statistic is the sample mean $\hat{\theta} = s(\mathbf{X}) = \frac{1}{n}\sum_{i=1}^{n} X_i$.

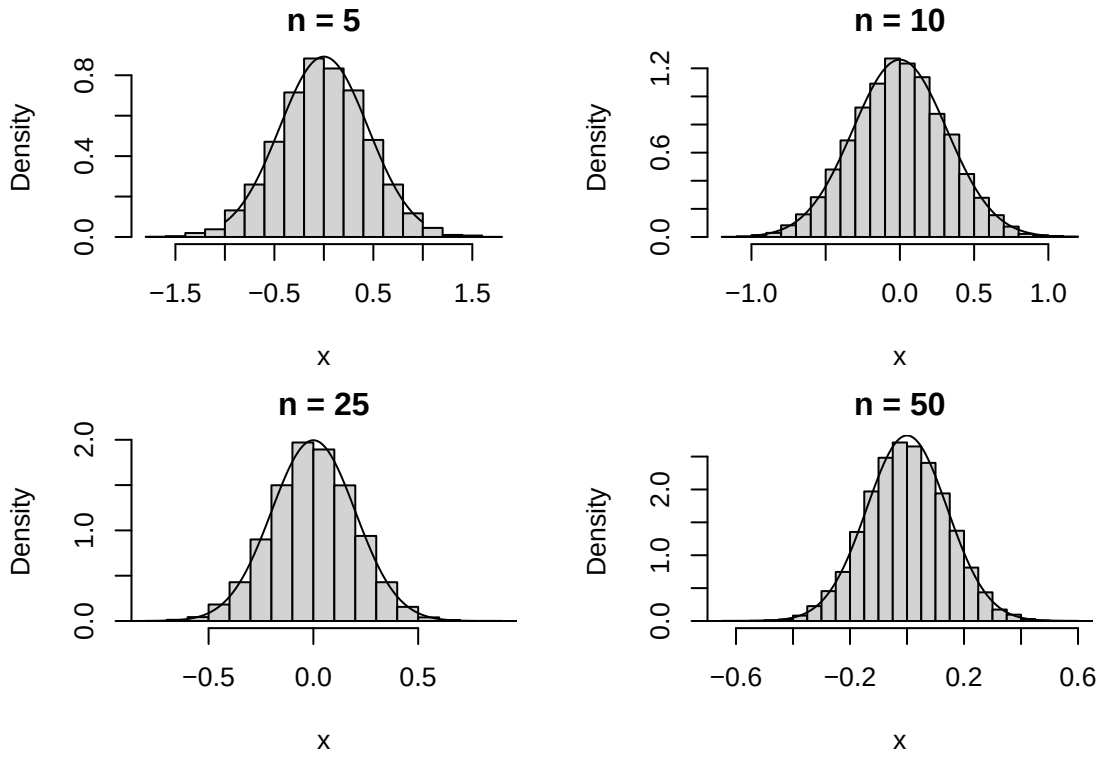
If the data generating distribution is Gaussian, i.e., $X \sim N(\theta, \sigma^2)$, then the sampling distribution is Gaussian, i.e., $\hat{\theta} \sim N(\theta, \sigma^2/n)$.

```
#loop through different n values
par(mfrow = c(2, 2), mar = c(4, 4, 2, 2))
for(n in c(5, 10, 25, 50)){

  #generate 10000 means calculated from N(0,1) data
  set.seed(1)
  x <- replicate(10000, mean(rnorm(n)))

  # plot histogram
  hist(x, freq = FALSE, main = paste0("n = ", n), breaks = 20)

  #add sampling distribution
  xseq <- seq(-1, 1, length.out = 1000)
  lines(xseq, dnorm(xseq, sd = 1 / sqrt(n)))
}
```



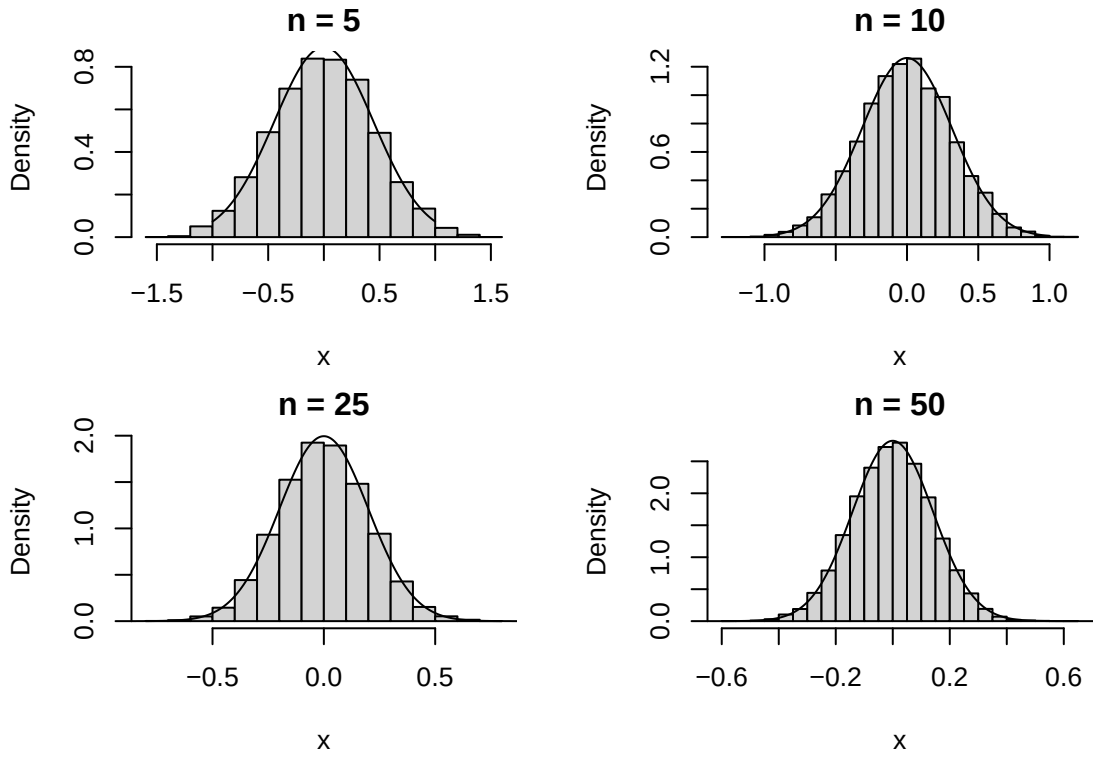
If the data generating distribution is some non-Gaussian distribution, then the sampling distribution of $\hat{\theta}$ is asymptotically Gaussian, i.e., $G(\theta) \to N(\theta, \sigma^2/n)$ as $n \to \infty$.

```r
#loop through different n values
par(mfrow = c(2, 2), mar = c(4, 4, 2, 2))
for(n in c(5, 10, 25, 50)){

  #generate 10000 means calculated from uniform data
  set.seed(1)
  x <- replicate(10000, mean(runif(n, min = -sqrt(3), max = sqrt(3))))

  #plot histogram
  hist(x, freq = FALSE, main = paste0("n = ", n), breaks = 20)

  #add sampling distribution
  xseq <- seq(-1, 1, length.out = 1000)
  lines(xseq, dnorm(xseq, sd = 1 / sqrt(n)))
}
```



Example 2: Median

Suppose that the parameter is the population median $P(X < \theta) = 1/2$, and the statistic is the sample mean $\hat{\theta} = \text{median}(\mathbf{x})$.

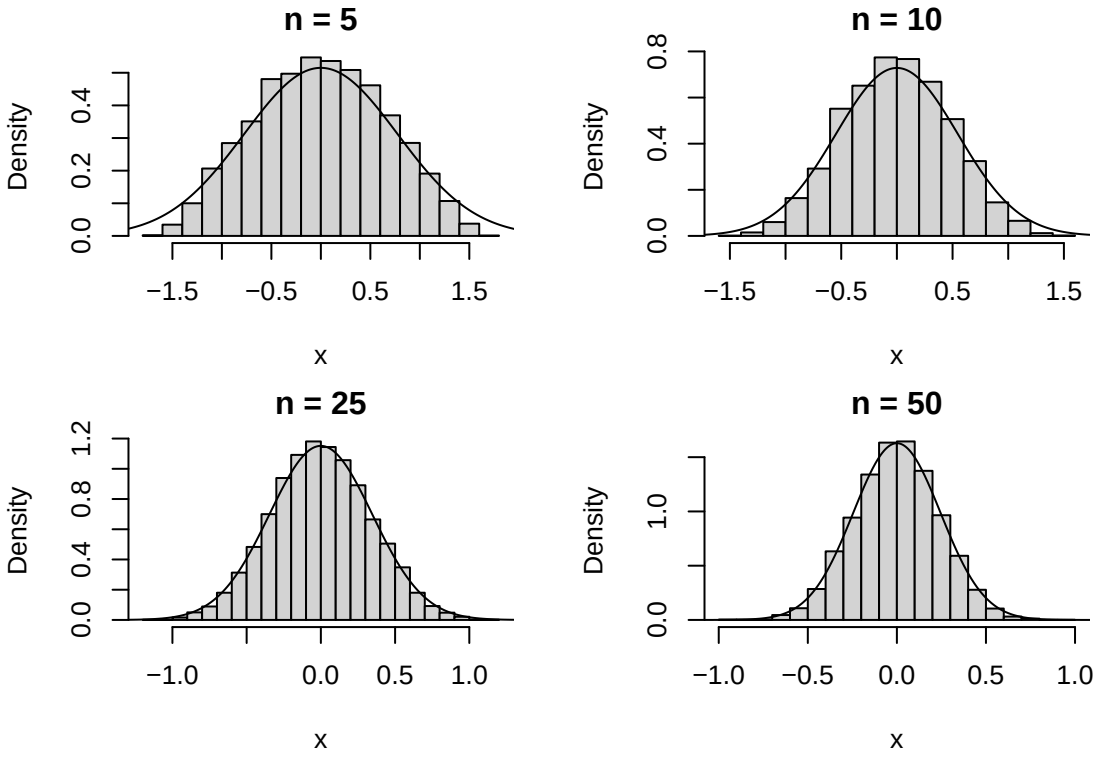
As $n \to \infty$, the sampling distribution of $\hat{\theta}$ approaches a normal distribution with mean θ and variance $\frac{1}{4nf(\theta)^2}$, where $f(x) = \frac{d}{dx}F(x)$ denotes the data-generating probability density function.

```r
#loop through different n values
par(mfrow = c(2, 2), mar = c(4, 4, 2, 2))
for(n in c(5, 10, 25, 50)){
```

```r
  #generate 10000 medians calculated from uniform data
  set.seed(1)
  x <- replicate(10000, median(runif(n, min = -sqrt(3), max = sqrt(3))))

  #plot histogram
  hist(x, freq = FALSE, main = paste0("n = ", n), breaks = 20)

  #add sampling distribution
  xseq <- seq(-2, 2, length.out = 1000)
  asymp.se <- 1 / sqrt(4 * n * dunif(0, min = -sqrt(3), max = sqrt(3))^2)
  lines(xseq, dnorm(xseq, sd = asymp.se))
}
```



1.4 Need for Nonparametric Bootstrap

The previous **example with the median reveals the need for the nonparametric bootstrap.** Specifically, for many statistics, there is no theoretical result for the exact sampling distribution, so statistical inference may only be possible using large sample approximations. And, in many cases (such as with the sample median), even the asymptotic distribution requires knowledge about the data generating distribution that is not readily obtainable in real data situations. Note that the asymptotic variance of the median requires knowledge of $f(\theta)$, which would rarely be known in any real data situation. Thus, for statistical inference about any generic parameter (i.e., any parameter that is not the mean), some method is needed that can produce a reasonable approximation of the sampling distribution — which is needed to conduct statistical inference.

2 Nonparametric Bootstrap Basics

2.1 Bootstrapping Procedure

Suppose that we have an observed sample of data $\mathbf{x} = (x_1, \ldots, x_n)^t$ with $X_i \overset{iid}{\sim} F$ for some distribution F. Furthermore, suppose that $\theta = T(F)$ is the parameter of interest, and $\hat{\theta} = s(\mathbf{X})$ is the statistic used to estimate θ from the sample of data.

The nonparametric bootstrap procedure is a rather simple idea:

1. Independently sample x_i^* with replacement from $\{x_1, \ldots, x_n\}$ for $i = 1, \ldots, n$.
2. Calculate the statistic $\hat{\theta}^* = s(\mathbf{x}^*)$ where $\mathbf{x}^* = (x_1^*, \ldots, x_n^*)^t$ is the resampled data.
3. Repeat steps 1-2 a total of R times to form the bootstrap distribution of $\hat{\theta}$.

The bootstrap distribution consists of R estimates of θ plus the original estimate $\hat{\theta}$, i.e., the bootstrap distribution is $\left\{\hat{\theta}, \hat{\theta}_1^*, \ldots, \hat{\theta}_R^*\right\}$. This bootstrap distribution can be used as a surrogate for the sampling distribution of $\hat{\theta}$ for the purpose of statistical inference. As will be demonstrated in the following sections, the bootstrap distribution can be used for estimating properties of $\hat{\theta}$ (e.g., standard error and bias), as well as for forming confidence intervals for θ.

Note: The number of replications should be rather large, e.g., $R \geq 9999$, to ensure that any calculations from the bootstrap distribution are not subject to substantial Monte Carlo error. The `np.boot` function (in the `nptest` package) uses a default of $R = 9999$ resamples to form the bootstrap distribution, but it may be desirable to increase this value. See the `mcse()` function (in the `nptest` package) for information about how the number of resamples R relates to the Monte Carlo standard error of the result.

2.2 Empirical Distribution

Definition

Suppose that we have an observed sample of data $\mathbf{x} = (x_1, \ldots, x_n)^t$ with $X_i \overset{iid}{\sim} F$ for some distribution F. The empirical cumulative distribution function (ecdf) uses the sample of data to estimate the unknown cdf F. The ecdf is defined as

$$\hat{F}_n(x) = \hat{P}_n(X \leq x) = \frac{1}{n} \sum_{i=1}^{n} I(x_i \leq x)$$

where $I()$ is an indicator function. Note that the ecdf assigns probability $1/n$ to each observation x_i for $i \in \{1, \ldots, n\}$, which implies that

$$\hat{P}_n(A) = \frac{1}{n} \sum_{i=1}^{n} I(x_i \in A)$$

for any set A in the sample space of X.

Since the ecdf is a proportion estimate from an iid sample of observations, we have that

$$E\left[\hat{F}_n(x)\right] = F(x)$$

which reveals that the ecdf is unbiased, and

$$\text{Var}\left(\hat{F}_n(x)\right) = \frac{1}{n} F(x)(1 - F(x))$$

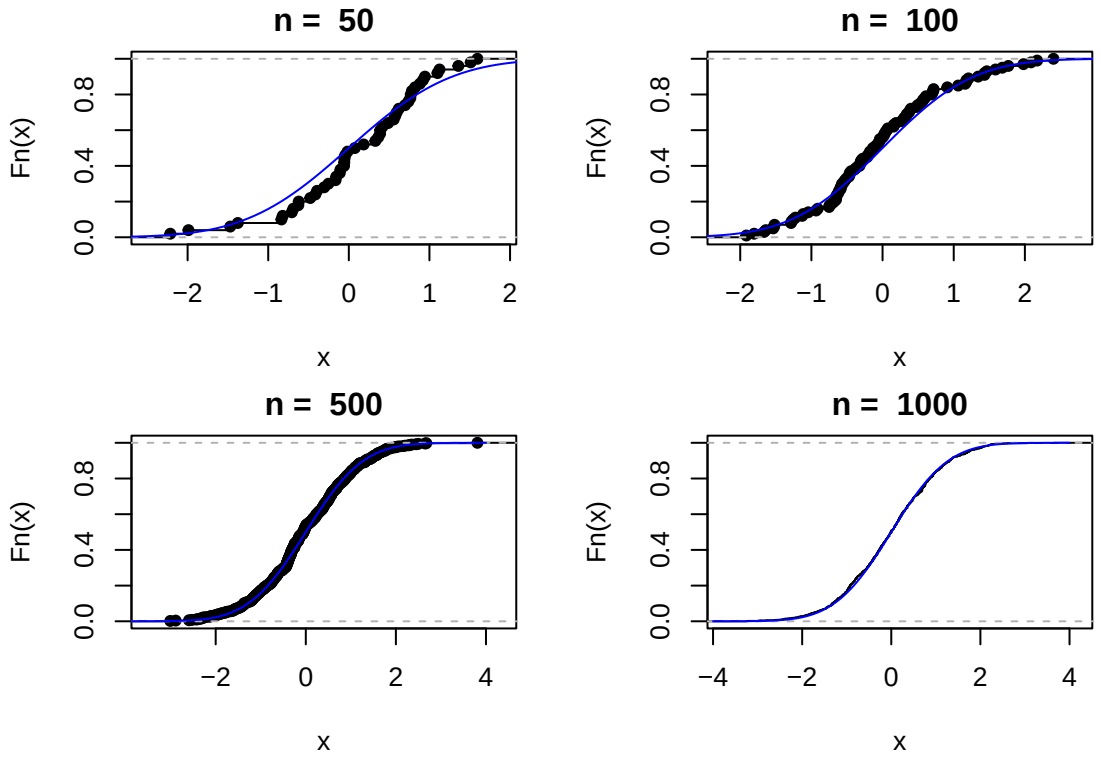
which reveals that the variance of the ecdf decreases as n increases. Furthermore, the Glivenko-Cantelli theorem reveals that as $n \to \infty$, we have that

$$\sup_{x \in \mathbb{R}} \left| \hat{F}_n(x) - F(x) \right| \xrightarrow{as} 0$$

where the notation $\xrightarrow{as}$ denotes almost sure convergence.

Example 1: Normal Distribution

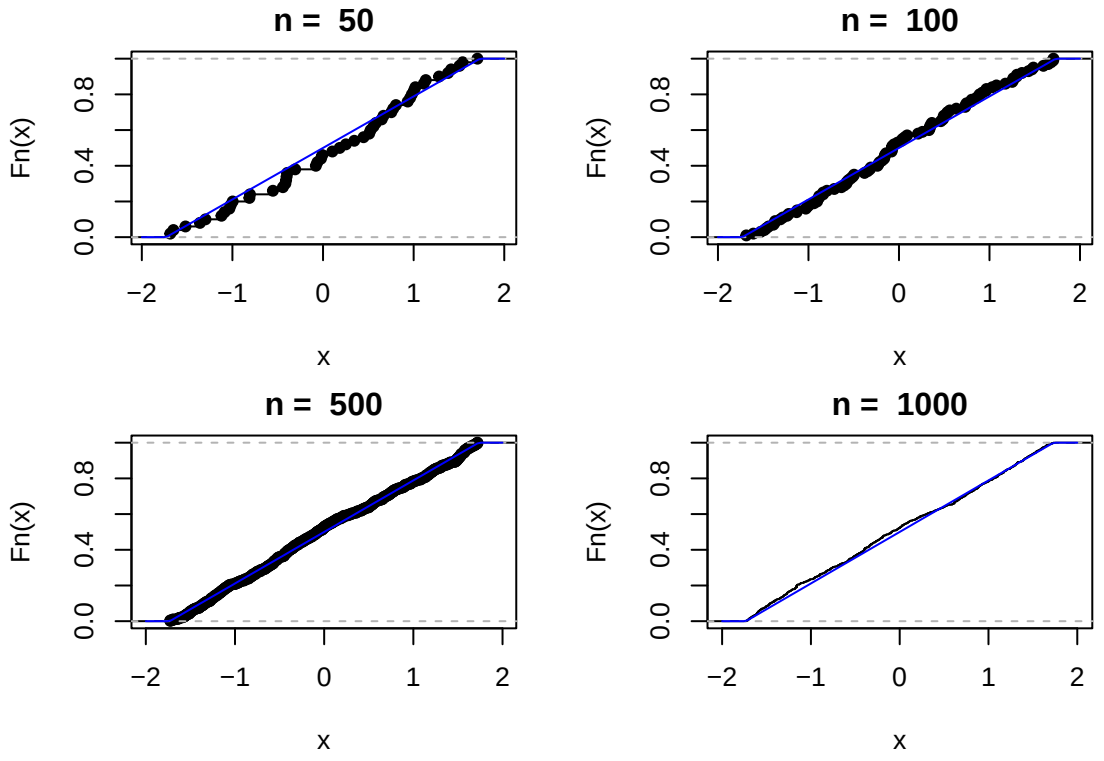
```
set.seed(1)
par(mfrow = c(2,2), mar = c(4, 4, 2, 2))
n <- c(50, 100, 500, 1000)
xseq <- seq(-4, 4, length.out = 100)
for(j in 1:4){
  x <- rnorm(n[j])
  plot(ecdf(x), main = paste("n = ",n[j]))
  lines(xseq, pnorm(xseq), col = "blue")
}
```



Example 2: Uniform Distribution

```
set.seed(1)
par(mfrow = c(2,2), mar = c(4, 4, 2, 2))
n <- c(50, 100, 500, 1000)
```

```r
xseq <- seq(-2, 2, length.out = 100)
for(j in 1:4){
  x <- runif(n[j], min = -sqrt(3), max = sqrt(3))
  plot(ecdf(x), main = paste("n = ",n[j]))
  lines(xseq, punif(xseq, min = -sqrt(3), max = sqrt(3)), col = "blue")
}
```



Example 3: Bivariate Data

Consider the following data on average LSAT scores and GPAs for $n = 15$ law schools. Note that these data represent a random sample of $n = 15$ law schools from a total collection of $N = 82$ American law schools that participated in the study.

School	LSAT (y)	GPA (z)	School	LSAT (y)	GPA (z)
1	576	3.39	9	651	3.36
2	635	3.30	10	605	3.13
3	558	2.81	11	653	3.12
4	578	3.03	12	575	2.74
5	666	3.44	13	545	2.76
6	580	3.07	14	572	2.88
7	555	3.00	15	594	2.96
8	661	3.43			

Table 1: Average LSAT scores and GPAs for $n = 15$ law schools (Efron & Tibshirani, 1993).

Suppose we want to estimate the proportion of schools that "safety schools", i.e., those schools that have an average LSAT score below 600 and an average GPA below 3. Defining $A = \{(y, z) : 0 < y < 600, 0 < z <$

3.00}, we can use the ecdf idea to estimate this proportion:

$$\hat{P}_{15}(A) = \frac{1}{15}\sum_{i=1}^{15} I((y_i, z_i) \in A) = 5/15 = 1/3$$

2.3 Plug-In Principle

Definition

Suppose that we have an observed sample of data $\mathbf{x} = (x_1, \ldots, x_n)^t$ with $X_i \overset{iid}{\sim} F$ for some distribution F. Furthermore, suppose that $\theta = T(F)$ is the parameter of interest. As a reminder, the notation $\theta = T(F)$ denotes that the parameter is a function $T()$ of the distribution function. The *plug-in estimate* of the parameter θ is defined as

$$\hat{\theta}_n = T(\hat{F}_n)$$

which applies the parameter defining function $T()$ to the ecdf in place of the cdf. Note that the plug-in estimate $\hat{\theta}_n$ does not necessarily have to be equivalent to the estimate of the parameter $\hat{\theta} = s(\mathbf{x})$ that would typically be used in practice.

Example 1: Mean

As a first example, suppose that the parameter is the population mean, i.e., $\theta = E_F[X] = \int x f(x) dx$, where $E_F[]$ denotes the expectation with respect to F. The plug-in estimate of θ has the form

$$\hat{\theta}_n = E_{\hat{F}_n}[X] = \sum_{i=1}^{n} x_i \hat{f}_i = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{x}$$

which calculates the expectation with respect to $\hat{F}_n$ in place of F. Note that $\hat{f}_i = 1/n$ is the probability that the ecdf assigns to the i-th observation. In this case, the plug-in estimate is the same as the estimate $\hat{\theta} = s(\mathbf{x}) = \bar{x}$ that would typically be used in practice.

Example 2: Variance

As a second example, suppose that the parameter is the population variance, i.e.,

$$\theta = \text{Var}(X) = E_F[(X - \mu)^2] = E_F\left[X^2\right] - (E_F[X])^2$$

where $\mu = E_F[X]$ is the expected value of X. The plug-in estimate of the variance has the form

$$\hat{\theta}_n = E_{\hat{F}_n}\left[X^2\right] - (E_{\hat{F}_n}[X])^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 - \left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

which replaces the expectation operator $E_F[]$ with the empirical expectation operator $E_{\hat{F}_n}[]$. In this case, the plug-in estimate of the variance is equivalent to the maximum likelihood estimator of the variance (with a divisor of n). Note that this differs from the unbiased estimator

$$\hat{\theta} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

which is typically preferred over the maximum likelihood estimator $\hat{\theta}_n$.

2.4 Logic of the Bootstrap

For conducting statistical inference with unknown probability distributions, the nonparametric bootstrap treats the ecdf $\hat{F}_n$ as if it were the true cdf F. In other words, the nonparametric bootstrap treats the observed sample $\mathbf{x} = (x_1, \ldots, x_n)^t$ as if it were the true population. As a reminder, the sampling distribution of $\hat{\theta} = s(\mathbf{x})$ is the distribution of the $\hat{\theta}$ values that would be observed if the statistic function $s()$ was applied to a large number of independent samples of data from the population. The nonparametric bootstrap approximates the sampling distribution by applying the statistic function $s()$ to a large number of independent samples of data from the observed sample. As a result, **the nonparametric bootstrap can ONLY well approximate the sampling distribution of a statistic if the ecdf is a good estimate of the unknown data generating cdf**. This will be the case if the sample size n is large, but is not guaranteed (or even likely) for small samples of data.

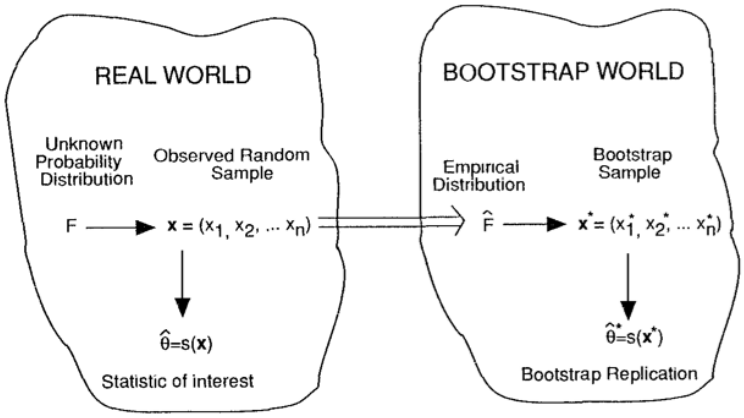


Figure 1: Illustration of the logic of bootstrap (Efron & Tibshirani, 1993).

3 Nonparametric Bootstrap Applications

3.1 Assessing the Quality of Estimators

Variance and Standard Error

The variance of an estimator $\hat{\theta} = s(\mathbf{x})$ for some parameter $\theta = T(F)$ is defined as

$$\text{Var}(\hat{\theta}) = E_F\left[(\hat{\theta} - E_F[\hat{\theta}])^2\right] = E_F\left[\hat{\theta}^2\right] - (E_F[\hat{\theta}])^2$$

where $E_F[]$ denotes that the expectation is calculated with respect to the data generating distribution F. The nonparametric bootstrap can be used to estimate the variance (or standard error) of an estimator $\hat{\theta} = s(\mathbf{x})$. Given the bootstrap distribution $\left\{\hat{\theta}, \hat{\theta}_1^*, \ldots, \hat{\theta}_R^*\right\}$, the bootstrap estimate of the variance has the form

$$\widehat{\text{Var}}(\hat{\theta}) = \frac{1}{R}\sum_{r=0}^{R}\left(\hat{\theta}_r^* - \bar{\theta}^*\right)^2$$

where $\hat{\theta}_0^* = \hat{\theta}$ is the estimate of θ from the observed sample and $\bar{\theta}^* = \frac{1}{R+1}\sum_{r=0}^{R}\hat{\theta}_r^*$ is the sample mean of the bootstrap distribution. Note that the estimated variance $\widehat{\text{Var}}(\hat{\theta})$ is the sample variance of the bootstrap distribution. The corresponding estimate of the standard error has the form

$$\widehat{SE}(\hat{\theta}) = \sqrt{\widehat{\text{Var}}(\hat{\theta})} = \sqrt{\frac{1}{R}\sum_{r=0}^{R}\left(\hat{\theta}_r^* - \bar{\theta}^*\right)^2}$$

which is the sample standard deviation of the bootstrap distribution. Note that as the number of bootstrap samples gets infinitely large, i.e., as $R \to \infty$, the bootstrap estimate of the standard error converges to the plug-in estimate of the standard error of $\hat{\theta}$.

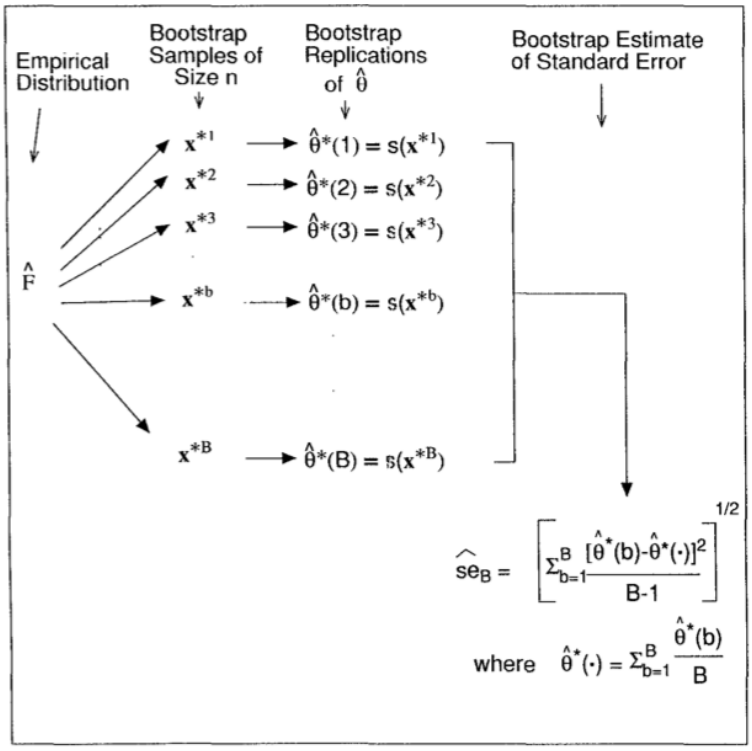


Figure 2: A schematic of the bootstrap standard error estimation (Efron & Tibshirani, 1993).

Bias

The bias of an estimator $\hat{\theta} = s(\mathbf{x})$ for some parameter $\theta = T(F)$ is defined as

$$\text{Bias}(\hat{\theta}) = E_F[\hat{\theta}] - \theta$$

where $E_F[]$ denotes that the expectation is calculated with respect to the data generating distribution F. The plug-in estimate of the bias uses the ecdf $\hat{F}_n$ instead of the unknown true cdf F. Specifically, the plug-in estimate of the bias has the form

$$\widehat{\text{Bias}}_n(\hat{\theta}) = E_{\hat{F}_n}[s(\mathbf{X})] - \hat{\theta}_n$$

where $E_{\hat{F}_n}[]$ is the expectation calculated with respect to $\hat{F}_n$ and $\hat{\theta}_n = T(\hat{F}_n)$ is the plug-in estimate of θ. This implies that the bootstrap estimate of the bias can be calculated as

$$\widehat{\text{Bias}}(\hat{\theta}) = \bar{\theta}^* - \hat{\theta}$$

where $\bar{\theta}^* = \frac{1}{R+1}\sum_{r=0}^{R} \hat{\theta}_r^*$ is the sample mean of the bootstrap distribution. Note that this assumes that $\hat{\theta} = T(\hat{F}_n)$, i.e., that the statistic is the plug-in estimate of θ. If $\hat{\theta}$ is not the plug-in estimate, then $\hat{\theta}$ should be replaced by $T(\hat{F}_n)$ in the definition of $\widehat{\text{Bias}}(\hat{\theta})$.

Mean Squared Error

The mean squared error (MSE) of an estimator $\hat{\theta} = s(\mathbf{x})$ for some parameter $\theta = T(F)$ is defined as

$$\text{MSE}(\hat{\theta}) = E_F[(\hat{\theta} - \theta)^2] = \text{Var}(\hat{\theta}) + \text{Bias}(\hat{\theta})^2$$

where $E_F[]$ denotes that the expectation is calculated with respect to the data generating distribution F. The bootstrap estimate of the MSE involves calculating the expectation using the ecdf $\hat{F}_n$ in place of the unknown true cdf F. Note that since the MSE can be decomposed into an additive function of the variance and bias, the bootstrap estimate of the MSE simply involves adding the bootstrap estimate of the variance and (squared) bias.

3.3 Univariate Data Examples

Overview

Univariate samples of data have the form $\mathbf{x} = (x_1, \ldots, x_n)^t$ where each $X_i \overset{iid}{\sim} F$ for some distribution F. For univariate data, using the `np.boot` function is rather simple, given that it just requires inputting the data vector `x` and the statistic function statistic (and possibly additional arguments passed through the ... argument). In the following examples, I will demonstrate how to use the `np.boot` function with univariate data using predefined functions in `R` for the statistic function. However, it should be noted that the statistic argument can be a user-defined function, which makes it possible to calculate any statistic that is of interest.

Example 1: Univariate Statistic (Median)

For this example, we will generate $n = 100$ observations from a standard normal distribution, and use the median as the parameter/statistic of interest. Note that the true (population) median is zero. Since the median is a univariate statistic, the bootstrap distribution will be a vector of length $R + 1$ containing the bootstrap replicates of the median.

```
#generate 100 standard normal observations
set.seed(1)
n <- 100
x <- rnorm(n)

#nonparametric bootstrap
npbs <- np.boot(x = x, statistic = median)
npbs
```

```
##
## Nonparametric Bootstrap of Univariate Statistic
## using R = 9999 bootstrap replicates
##
##   t0: 0.1139
##   SE: 0.1394
## Bias: 0.0185
##
## BCa Confidence Intervals:
##       lower  upper
## 90% -0.0566 0.3411
## 95% -0.0811 0.3673
## 99% -0.1351 0.3940
```

```
median(x)                          # t0
```

```
## [1] 0.1139092
```

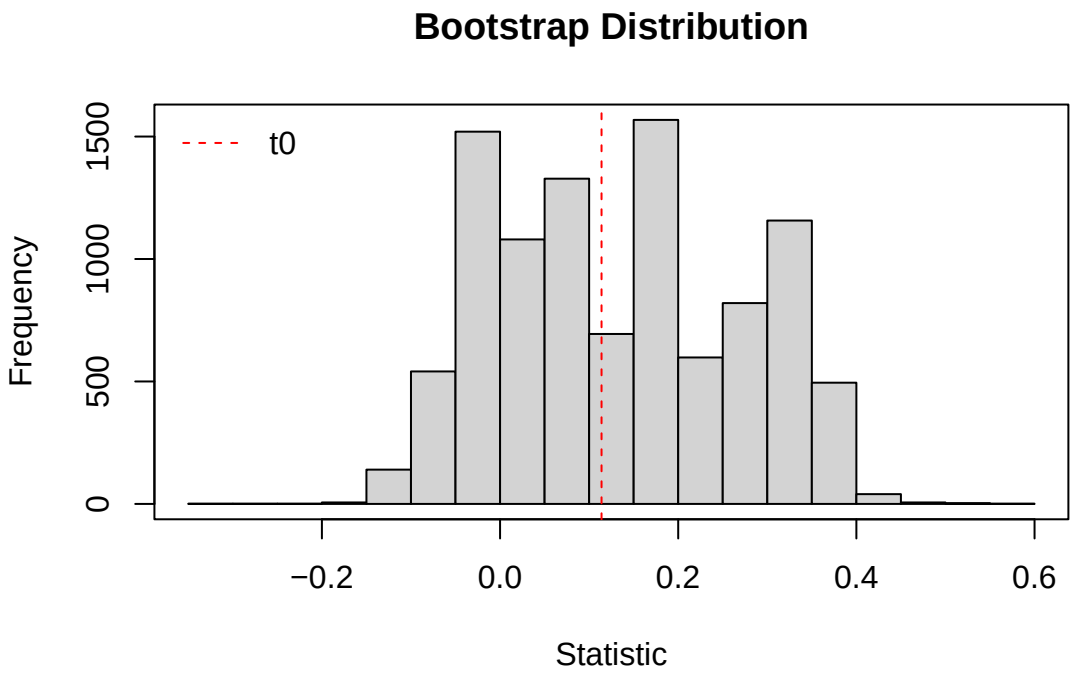
```r
sd(npbs$boot.dist)                  # SE
```

```
## [1] 0.1393567
```

```r
mean(npbs$boot.dist) - npbs$t0      # Bias
```

```
## [1] 0.01845341
```

```r
hist(npbs$boot.dist, xlab = "Statistic", main = "Bootstrap Distribution")
box()
abline(v = npbs$t0, lty = 2, col = "red")
legend("topleft", "t0", lty = 2, col = "red", bty = "n")
```



2: Multivariate Statistic (Quartiles)

For this example, we will generate $n = 100$ observations from a standard normal distribution, and use the quartiles as the parameters/statistics of interest. Note that the true (population) quartiles are $Q_1 =$ `qnorm(0.25) = -0.6744898`, $Q_2 =$ `qnorm(0.5) = 0`, and $Q_3 =$ `qnorm(0.75) = 0.6744898`. Since the quartiles are a multivariate statistic, the bootstrap distribution will be a matrix of dimension $(R + 1) \times 3$, where each column contains the bootstrap replicates of the corresponding quartile.

```r
#generate 100 standard normal observations
set.seed(1)
n <- 100
x <- rnorm(n)
```

```r
#nonparametric bootstrap (using ... to enter 'probs' argument)
npbs <- np.boot(x = x, statistic = quantile,
                probs = c(0.25, 0.5, 0.75))
npbs
```

```
##
## Nonparametric Bootstrap of Multivariate Statistic
## using R = 9999 bootstrap replicates
##
##           25%    50%     75%
##   t0: -0.4942 0.1139  0.6915
##   SE:  0.1172 0.1394  0.0933
## Bias:  0.0058 0.0185 -0.0170
##
## 95% BCa Confidence Intervals:
##           25%     50%    75%
## lower -0.6941 -0.0811 0.5047
## upper -0.2534  0.3673 0.8811
```

```r
quantile(x, probs = c(0.25, 0.5, 0.75))     # t0
```

```
##        25%        50%        75%
## -0.4942425  0.1139092  0.6915454
```

```r
apply(npbs$boot.dist, 2, sd)                # SE
```

```
##        25%        50%        75%
## 0.11724637 0.13935672 0.09333269
```

```r
colMeans(npbs$boot.dist) - npbs$t0          # Bias
```

```
##          25%          50%          75%
##  0.005771337  0.018453409 -0.017043706
```

```r
npbs$cov
```

```
##             25%         50%         75%
## 25% 0.013746712 0.009523836 0.003761904
## 50% 0.009523836 0.019420295 0.007248275
## 75% 0.003761904 0.007248275 0.008710991
```

```r
cov(npbs$boot.dist)
```

```
##             25%         50%         75%
## 25% 0.013746712 0.009523836 0.003761904
## 50% 0.009523836 0.019420295 0.007248275
## 75% 0.003761904 0.007248275 0.008710991
```

```
par(mfrow = c(1,3))
for(j in 1:3){
  hist(npbs$boot.dist[,j], xlab = "Statistic",
       main = paste0("Bootstrap Distribution", ": Q", j))
  box()
  abline(v = npbs$t0[j], lty = 2, col = "red")
  legend("topright", paste0("t0[",j,"]"), lty = 2, col = "red", bty = "n")
}
```