Teaching Bootstrap Methods with R Programming

Xuemao Zhang
USCOTS 2021
East Stroudsburg University

Overview

- Why should the method of bootstrap be taught by programming?
  - *Bootstrap is a resampling method*
  - *Programming can help students solve complicated problems*
- Teaching Examples
  - *A simulation study: Estimation of a single population mean*
  - *Estimating coefficients in simple linear regression models*
  - *Block bootstrap for time-series data*

Bootstrap method

- Bootstrap is a **resampling** method: drawing repeated samples of size k with replacement from the original data of size k

- The bootstrap is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator and is particularly good for getting their:

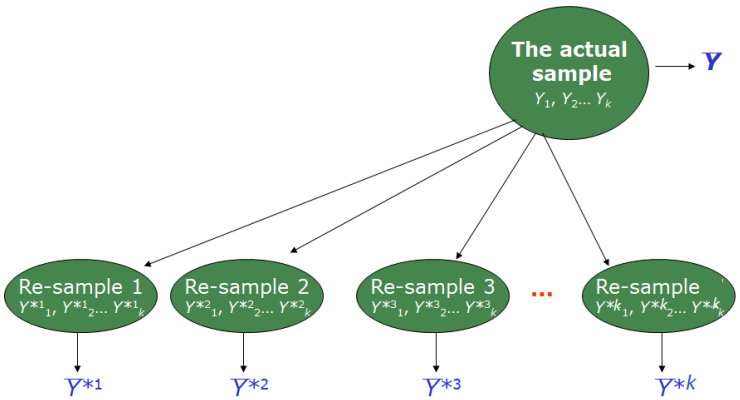
  - *standard errors*
  - *and confidence limits*

- The use of the term bootstrap derives from the phrase , widely thought to be based on one of the eighteenth century “Surprising Adventures of Baron Munchausen” by Rudolph Erich Raspe:

```
The Baron had fallen to the bottom of a deep lake.
Just when it looked like all was lost, he thought to
pick himself up by his own bootstraps.
```

Bootstrap method

- Treat the data of size k as a population (a proxy for the true distribution).
- And sample with replacement k times.
- Each resample simulates the process of taking a sample from the “true” distribution. Compute the statistic of interest on each “re-sample”.
- $\{\overline{Y^*}\}$ constitutes an estimate of the distribution of $\overline{Y}$.



Bootstrap method

- **Idea**: We want to find the confidence interval for a mean from a sample of only 4 observations: 6, -3, 5, 3.

If the data are from a normal population, we use the general t-interval to get a confidence interval of the population mean.

- The first thing that bootstrapping does is estimate the population distribution of Y from the four observations in the sample
- In other words, the random variable Y^* is defined:

y^*	$p^*(y^*)$
6	0.25
-3	0.25
5	0.25
3	0.25

- The mean of Y^* is then simply the mean of the sample:

$$E^*(Y^*) = \sum y^* p^*(y^*) = 2.75 = \overline{Y}$$

with population variance

$$Var^*(Y^*) = \sum (y^* - 2.75)^2 p^*(y^*) = 12.1875.$$

Bootstrap method

- We now treat the sample as if it were the population, and resample from it
- In this example we take all possible samples with replacement, meaning that we take $k^k = 4^4 = 256$ different samples
- Since we found all possible samples, the mean of these samples is simply the original mean
- The variance of the sample means from these samples is:

$$Var^*(\overline{Y}^*) = \frac{\sum_{b=1}^{k^k}(\overline{Y_b}^* - \overline{Y})^2}{k^k} \approx 3.047$$

Bootstrap method

- Some of the original data points will appear more than once; others won't appear at all.
- For a sample of size 100, if we sample with replacement 100 times. In fact, there is a chance of

$$(1 - 1/100)^{100} \approx 0.366$$

that any one of the original data points won't appear at all.

  - *any data point is included with Prob* ≈ 0.634.

- We treat the original sample as the "true population".
- Each resample simulates the process of taking a sample from the "true" distribution.

Estimation of a single population mean

- Sampling 100 data points from $N(\mu = 50, \sigma = 10)$

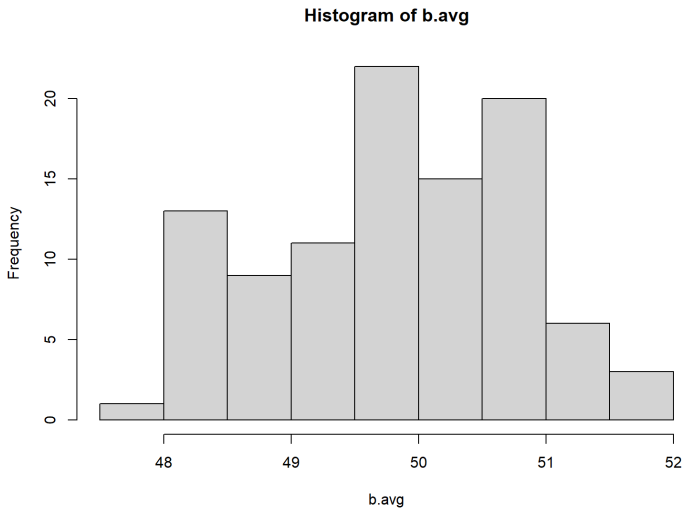
```
norm.data <- rnorm(100, mean=50, sd=10);
```

- Bootstrap method to estimate μ

```
set.seed(20);
b.avg = c();
sd.est=c();
for(i in 1:100)
{
ystar=sample(norm.data,length(norm.data),replace=T);
b.avg[i] = mean(ystar);
sd.est[i]=sd(ystar);
}
```

Estimation of a single population mean

```
hist(b.avg)
```



Estimation of a single population mean

- By the sampling theory

$$\overline{Y} \sim N(\mu_{\overline{Y}} = 50, \sigma_{\overline{Y}} = \frac{10}{\sqrt{n}} = 1)$$

```
mean(b.avg)
```

```
## [1] 49.83196
```

```
sd(b.avg)
```

```
## [1] 0.9564634
```

Estimation of a single population mean

Confidence Interval estimation

- The first way is by the approximate normality assumption. For example, a 95% confidence interval of μ is

```
c(mean(b.avg)-1.96*sd(b.avg), mean(b.avg)+1.96*sd(b.avg));
```

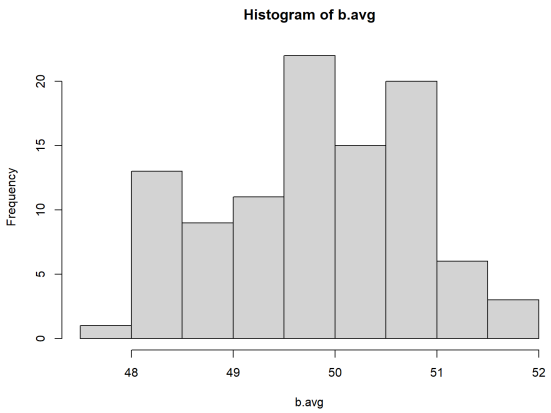
```
## [1] 47.95729 51.70663
```

- The second is to use the percentiles of the bootstrap histogram. A 95% confidence interval of μ is

```
quantile(b.avg, c(0.025, 0.975));
```

```
##     2.5%    97.5%
## 48.23349 51.54375
```

```
hist(b.avg)
```



Estimation of a single population mean

boot::boot()

- Performing a bootstrap analysis in using the `boot` library.
  - *First, we must create a function that computes the statistic of interest for a bootstrap sample indicated by `index`.*
  - *Second, we use the `boot()` function, which is part of the `boot` library, to perform the bootstrap by repeatedly sampling observations from the data set with replacement.*

```
mean.fn=function(data,index){  return(mean(data[index])); }
library(boot);
boot(norm.data, mean.fn, R=100);
```

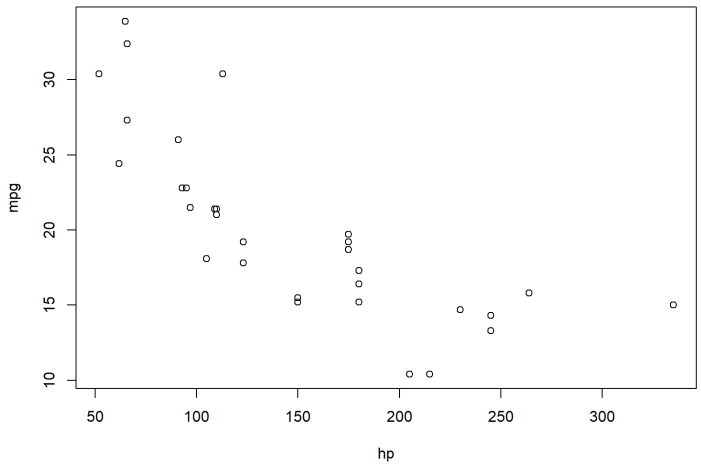
```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = norm.data, statistic = mean.fn, R = 100)
##
##
## Bootstrap Statistics :
##     original     bias    std. error
## t1* 49.81992 -0.1306911    0.950393
```

```
#set.seed(1);
#mean.fn(norm.data,sample(100,100,replace=T));
```

Estimating coefficients in simple linear regression models

- Consider the `mtcars` data set and use hp(horsepower) to predict `mpg`.

```
plot(mpg~hp,data=mtcars)
```



Estimating coefficients in simple linear regression models

- We first create a simple function, `boot.fn()`, which takes in the `mtcars` data set as well as a set of indices for the observations, and returns the intercept and slope estimates for the linear regression model.

```
boot.fn=function(data,index)
 return(coef(lm(mpg~hp,data=data,subset=index)))
```

```
boot.fn(mtcars,1:32);
```

```
## (Intercept)          hp
## 30.09886054 -0.06822828
```

```
set.seed(1)
boot.fn(mtcars,sample(32,32,replace=T));
```

```
## (Intercept)          hp
## 29.38901482 -0.06886018
```

Estimating coefficients in simple linear regression models

- Next, we use the `boot()` function to compute the standard errors of 32 bootstrap estimates for the intercept and slope terms.

```
boot(mtcars,boot.fn,32);
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = mtcars, statistic = boot.fn, R = 32)
##
##
## Bootstrap Statistics :
##        original       bias    std. error
## t1* 30.09886054  0.567331366   1.9896507
## t2* -0.06822828 -0.004345474   0.0141649
```

- Compare the bootstrap estimates with the general linear regression model fit.
  - *The standard errors might be under-estimated*

```
summary(lm(mpg~hp,data=mtcars))$coef;
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 30.09886054  1.6339210 18.421246 6.642736e-18
## hp          -0.06822828  0.0101193 -6.742389 1.787835e-07
```

Estimating coefficients in simple linear regression models

- We also can consider the bootstrap method applied to multiple linear regression model. For example, quadratic regression model.

```
boot.fn=function(data,index)
{coefficients(lm(mpg~hp+I(hp^2),
                 data=data,subset=index))
}
set.seed(1); boot(mtcars,boot.fn,392)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = mtcars, statistic = boot.fn, R = 392)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1* 40.4091172029  6.741016e-01 3.1844475707
## t2* -0.2133082599 -1.080736e-02 0.0416863327
## t3*  0.0004208156  3.731752e-05 0.0001221299
```

```
summary(lm(mpg~hp+I(hp^2),data=mtcars))$coef;
```

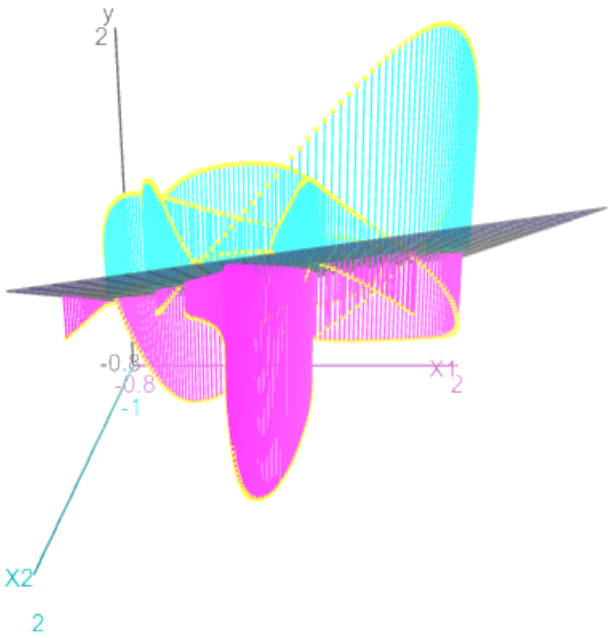
```
##                  Estimate   Std. Error   t value     Pr(>|t|)
## (Intercept) 40.4091172029 2.740759e+00 14.743766 5.234398e-15
## hp          -0.2133082599 3.488387e-02 -6.114812 1.162972e-06
## I(hp^2)      0.0004208156 9.844453e-05  4.274647 1.889240e-04
```

Block bootstrap for time-series data

- Suppose there is a time series data set.

```
library(readr);
Xy=read_csv("Xy.csv");
dim(Xy);
```

```
## [1] 1000    4
```



Block bootstrap for time-series data

- If we consider the general bootstrap method for the simple linear regression model.

```
boot.fn=function(data,index){
return( coef(lm(y~X1+X2,data=data,subset=index)));
}
boot(Xy, boot.fn, R=1000) #bootstrap 1000 times
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Xy, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 0.2658349 -1.770579e-05  0.01442441
## t2* 0.1453263  9.210938e-04  0.02844693
## t3* 0.3133670  1.750347e-03  0.03520836
```

Block bootstrap for time-series data

- Finally, use the **block bootstrap** to estimate $s.e.(\hat{\beta}_1)$. Use blocks of 100 contiguous observations, and resample ten whole blocks with replacement then paste them together to construct each bootstrap time series. For example, one of bootstrap resamples could be:

```
new.rows=c(101:200, 401:500, 101:200, 901:1000, 301:400,
             1:100, 1:100, 801:900, 201:300, 701:800);
new.Xy=Xy[new.rows, ];
dim(new.Xy);
```

```
## [1] 1000    4
```

```
head(new.Xy);
```

```
## # A tibble: 6 x 4
##       t    X1     X2     y
##   <dbl> <dbl>  <dbl> <dbl>
## 1   101 0.327 0.0549  1.14
## 2   102 0.359 0.0538  1.13
## 3   103 0.390 0.0537  1.13
## 4   104 0.420 0.0547  1.13
## 5   105 0.450 0.0568  1.12
## 6   106 0.478 0.0598  1.12
```

Block bootstrap for time-series data

- Now let's write the new bootstrap function.

```
newboot.fn=function(data, index){
  seq=sample(10,10,replace=T);#10 random blocks
  newrows=integer();
for (i in 1:10)
 {
newrows=c(newrows, sample(((seq[i]-1)*100+1):(seq[i]*100),
          100,replace=T) );
}
 index=newrows;
 return( coef(lm(y~X1+X2,data=data,subset=index)) );
}
```

Block bootstrap for time-series data

```
boot(Xy, newboot.fn, R=1000);
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Xy, statistic = newboot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##      original     bias    std. error
## t1* 0.2407740 0.03016918  0.09238234
## t2* 0.1072043 0.04308182  0.20017263
## t3* 0.3318433 0.06771230  0.32394562
```

```
boot(Xy, boot.fn, R=1000) #general bootstrap
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = Xy, statistic = boot.fn, R = 1000)
##
##
## Bootstrap Statistics :
##      original        bias    std. error
## t1* 0.2658349 -1.530411e-05  0.01451993
## t2* 0.1453263 -9.370238e-04  0.02899532
## t3* 0.3133670  2.112013e-05  0.03668959
```

Thank you!