

**22S:166****Introduction to the Bootstrap**

Lecture 8  
September 21, 2007

Kate Cowles  
374 SH, 335-0727  
kcowles@stat.uiowa.edu

**Resources**

- Efron, B. (1982) *The Jackknife, the Bootstrap, and Other Resampling Plans*. Number 38 in CBMS-NSF Regional Conference Series in Applied Mathematics. Philadelphia: SIAM.
- Efron, B. and Tibshirani, R.J. (1993) *An Introduction to the Bootstrap*. New York: Chapman & Hall.
- Davison, A.c. and Hinkley, D.V. (1997) *Bootstrap Methods and their Application*, New York: Cambridge University Press.

## Review concepts

- suppose we have one sample of  $n$  data values:  
 $y_1, \dots, y_n$
- sample values considered outcomes of i.i.d. random variables  $Y_1, \dots, Y_n$
- probability density function (pdf) or probability mass function (pmf)  $f$
- cumulative distribution function (cdf)  $F$
- sample will be used to make inference
  - about population characteristic  $\theta$
  - using statistic  $T$  whose value in sample is  $t$
- questions of interest regarding  $T$ 
  - bias?
  - standard error?
  - quantiles?
  - how to compute confidence limits for  $\theta$ ?

– likely values under a null hypothesis of interest?

## Two classes of statistical methods

- parametric
  - particular mathematical model for behavior of random variables  $Y_j$
  - pdf or pmf  $f$  is completely determined by values of unknown parameters  $\psi$
  - quantity of interest in statistical analysis  $\theta$  is a component or function of  $\psi$
- nonparametric
  - uses only the fact the  $Y_j$ s are i.i.d.
  - no mathematical model for their distribution
  - (may be useful to do a nonparameteric analysis even if a reasonable parametric model exists)
    - \* to assess sensitivity of conclusions to assumptions of parametric model

Example for the nonparametric bootstrap:  
City population data

- for each of  $n = 49$  U.S. cities, two data values
  - $u_j$  = population in 1920 (in 1000s)
  - $x_j$  = population in 1930 (in 1000s)
- population of interest is all U.S. cities
- the 49 cities are assumed to be a simple random sample from this population
- define  $(U, X)$  as pair of population values for a randomly selected city
- then if we knew  $\theta = \frac{E(X)}{E(U)}$  and the total 1920 population for the U.S., we could estimate the total 1930 population of U.S.
- want to estimate  $\theta$  without assuming any parametric model for  $X$  and  $U$
- sample-based statistic is  $T = \frac{\bar{X}}{\bar{U}}$

## The empirical distribution

- puts probability mass  $\frac{1}{n}$  at each sample value  $y_j$
- empirical distribution function (edf) or  $\hat{F}$ 
  - nonparametric mle of  $F$
  - sample proportion  $\hat{F}(y) = \frac{\#\{y_j \leq y\}}{n}$ 
    - \* where  $\#$  denotes the number of items in a set
- edf plays role of fitted model when no mathematical form is assumed for  $F$

- observations 1 to 10 of this dataset are included with the `boot` package for R

```
> data(city)
> city
      u  x
1  138 143
2   93 104
3   61  69
4  179 260
5   48  75
6   37  63
7   29  50
8   23  48
9   30 111
10   2  50
```

11

4. repeat step 2 independently a large number  $B$  of times obtaining bootstrap replications  $\hat{\theta}^{*1}, \hat{\theta}^{*2}, \dots, \hat{\theta}^{*B}$
5. Use bootstrap replications to:
  - estimate standard error of  $\hat{\theta}$
  - estimate bias
  - obtain confidence interval

## The non-parametric bootstrap

- goal: to get an idea of the sampling distribution of the statistic  $T$  under repeated sampling from the population of interest
- basic idea: our sample data gives us all the information we have about the whole population
- steps:
  1. calculate statistics of interest (call it  $\hat{\theta}$  from dataset as a whole)
  2. fit edf  $\hat{F}$
  3. Draw a “bootstrap sample” from  $\hat{F}$  and calculate statistic of interest on bootstrap sample
    - i.e., draw a sample of size  $n$  from original dataset **with replacement**
    - $Y_1^*, Y_2^*, \dots, Y_n^* \sim \hat{F}$
    - $\hat{\theta}^* = \hat{\theta}(Y_1^*, Y_2^*, \dots, Y_n^*)$

12

## Using the R sample function to draw bootstrap samples

sample package:base R Documentation

Random Samples and Permutations

Description:

'sample' takes a sample of the specified size from the elements of 'x' using either with or without replacement.

Usage:

```
sample(x, size, replace = FALSE, prob = NULL)
```

Arguments:

**x**: Either a (numeric, complex, character or logical) vector of more than one element from which to choose, or a positive integer.

**size**: non-negative integer giving the number of items to choose.

**replace**: Should sampling be with replacement?

**prob**: A vector of probability weights for obtaining the

elements of the vector being sampled.

```
> x <- seq(1:25)
> x
[1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
    19 20 21 22 23 24 25
> sample(x, 25)
[1]  2 20  3  9  6  8 15 10 23  1 19 25 12 21 14  4 13 24
    17  5 11 18  7 22 16
> sample(x, 25, replace = TRUE)
[1]  4  6 16 11 21 17  6 12  5  8 15 19 23 16 15 20 18 19
    21  5 25  7  8 20  3
```

## Bias correction using the bootstrap

- notation

- $\theta$  – true and unknown population quantity value
- $\hat{\theta}$  – estimate of  $\theta$  based on sample data
- $\hat{\theta}^{*b}$  – estimate of  $\theta$  from b-th bootstrap sample

## Bias correction continued

- So in a sense:
  - $\hat{\theta}^*$ s are to  $\hat{\theta}$  as  $\hat{\theta}$  is to  $\theta$
- bootstrap estimate of bias
  - Note: bias =  $E_F(\hat{\theta} - \theta)$

$$\begin{aligned} \widehat{bias}_{boot} &= \frac{1}{B} \left( \sum_{b=1}^B \hat{\theta}^{*b} - \hat{\theta} \right) \\ &= \hat{\theta}^{*\cdot} - \hat{\theta} \end{aligned}$$

- So bias-corrected point estimate is

$$\begin{aligned} \tilde{\theta} &= \hat{\theta} - (\hat{\theta}^{*\cdot} - \hat{\theta}) \\ &= 2\hat{\theta} - \hat{\theta}^{*\cdot} \end{aligned}$$

## Percentile method for confidence intervals

- denote cdf of *bootstrap distribution* of  $\hat{\theta}^*$  as
 
$$C\bar{D}F(t) = Pr_*(\hat{\theta}^* \leq t)$$
- If bootstrap distribution is obtained by simulation then

$$C\bar{D}F(t) \simeq \frac{\#(\hat{\theta}^{*b} \leq t)}{B}$$

- define confidence interval as interval between appropriate quantiles

## R code for the City Data

## Main driver program

```

> drive.bootstrap
function(mdata, mlefunc, bootfunc, B)
{
  mle <- mlefunc(mdata)          # compute mle
  print(c("mle",mle))
  boots <- bootfunc(mdata, B)    # generate bootstrap samples
  print("quantiles")
  print(quantile(boots, c(0.005, 0.025, 0.5, 0.975, 0.995)))
  meanb <- mean(boots)
  print(c("mean", meanb))
  stderr <- sqrt( var(boots) )
  print(c("stderr", stderr))
  biasc <- 2 * mle - meanb
  print("bias-corrected point estimate")
  biasc
}

```

```

}
bootratio
}

```

Function defining computation of  $\hat{\theta}$ 

```

> meanratio
function(mdata)
{
  mean(mdata$x)/mean(mdata$u)
}

```

## Function carrying out desired type of bootstrap

```

> nonparboot.ratio
function(mydat, B)
{
  # nonparametric bootstrap for ratio of means
  # data object must contain 2 columns of data
  # returns B bootstrap estimates of ratio of means
  if(ncol(mydat) != 2) {
    print("input matrix must have 2 columns of numeric data")
  }
  else {
    bootratio <- numeric()
    n <- nrow(mydat)      # number of observations
    for(i in 1:B) {
      index1 <- sample(n, replace = T)
      # sample of size n from integers 1:n with replacement
      boot1 <- mydat[index1, ]
      # bootstrap sample; rows of data corresponding to index1
      bootratio <- c(bootratio, mean(boot1$x) /
                    mean(boot1$u))
    }
  }
}

```

## Function call and results

```

library(boot)
data(city)
drive.bootstrap(city, meanratio, nonparboot.ratio, 81)

> drive.bootstrap(city, meanratio, nonparboot.ratio, 81)
[1] "mle"          "1.5203125"
[1] "quantiles"
      0.5%      2.5%      50%      97.5%      99.5%
1.229048 1.276364 1.523894 2.277978 2.651724
[1] "mean"          "1.58087829542998"
[1] "stderr"        "0.268075889571327"
[1] "bias-corrected point estimate"
[1] 1.459747

> drive.bootstrap(city, meanratio, nonparboot.ratio, 1000)
[1] "mle"          "1.5203125"
[1] "quantiles"
      0.5%      2.5%      50%      97.5%      99.5%
1.195834 1.254945 1.529401 2.118204 2.553725
[1] "mean"          "1.57122730135862"
[1] "stderr"        "0.239441130366229"
[1] "bias-corrected point estimate"
[1] 1.469398

```

```

> drive.bootstrap(city, meanratio, nonparboot.ratio, 25000)
[1] "mle"          "1.5203125"
[1] "quantiles"
      0.5%      2.5%      50%      97.5%      99.5%
1.190476 1.249741 1.518617 2.110503 2.479174
[1] "mean"          "1.55977878209242"
[1] "stderr"         "0.223036277313829"
[1] "bias-corrected point estimate"
[1] 1.480846

```

## Example for parametric bootstrap

- times between failures of air-conditioning equipment
- assume parametric model: observed values are sampled from an exponential distribution with parameter  $\lambda$

$$f(y_j) = \frac{1}{\lambda} \exp\left(-\frac{y_j}{\lambda}\right)$$

- data comes with **boot** package for R
- mean failure time  $\lambda$  is estimated by  $T = \bar{Y}$

```

>data(aircondit)
>aircondit
      hours
1         3
2         5
3         7
4        18
5        43

```

```

6      85
7      91
8      98
9     100
10     130
11     230
12     487

```

## Parametric bootstrap

1. estimate *parametric* mle  $\hat{F}$  of unknown F
  - i.e., get mles of parameters
2. Draw a “bootstrap sample” from  $\hat{F}$  and calculate statistic of interest on bootstrap sample
  - i.e., simulate data values from parametric model using mles as parameters
  - $Y_1^*, Y_2^*, \dots, Y_n^* \sim \hat{F}$
  - $\hat{\theta}^* = \hat{\theta}(Y_1^*, Y_2^*, \dots, Y_n^*)$
3. repeat step 2 independently a large number B of times obtaining bootstrap replications  $\hat{\theta}^{*1}, \hat{\theta}^{*2}, \dots, \hat{\theta}^{*B}$
4. Use bootstrap replications to:
  - estimate standard error of  $\hat{\theta}$
  - estimate bias
  - obtain confidence interval

## R code for parametric bootstrap for the air conditioning data

```
> parboot.logexpmean
function(mdata, B)
{
  # parametric bootstrap for log of exponential mean parm
  # data object must contain 1 column of data
  # returns B bootstrap estimates of log exponential mean parm
  if(ncol(mdata) != 1) {
    print("input data must have 1 column of numeric data")
  }
  else {
    mle <- logexpmean( mdata )
    bootlogexpmean <- numeric()
    n <- nrow(mdata)      # number of observations
    for(i in 1:B) {
      boot1 <- rexp( n, exp( - mle ) )
      # bootstrap sample; n random draws from exponential distribution
      # with mle from mdata as parameter
      bootlogexpmean <- c(bootlogexpmean,
        logexpmean( as.matx(boot1) ) )
    }
    bootlogexpmean
  }
}
```

```
> logexpmean
function(mdata)
{
  if(ncol(mdata) != 1) {
    print("input data must have 1 column of numeric data")
  }
  else {
    log( mean(mdata) )
  }
}
```

```
> drive.bootstrap2
function(mdata, mfunc, bootfunc, B)
{
  mle <- mfunc(mdata)      # compute mle
  print(c("mle",mle))
  boots <- bootfunc(mdata, B) # generate bootstrap samples
  print("quantiles")
  print(quantile(boots, c(0.005, 0.025, 0.5, 0.975, 0.995)))
  meanb <- mean(boots)
  print(c("mean", meanb))
  stderr <- sqrt( var(boots) )
  print(c("stderr", stderr))
  biasc <- 2 * mle - meanb
  print("bias-corrected point estimate")
  print(biasc)
  # now bias-corrected percentile method of C.I.
  # first get Pr(a bootstrap estimate is <= rhohat)
  cof <- length(boots[boots <= mle])/B
  print(c("cof", cof))
  z0 <- qnorm(cof) # corresponding normal quantile
  z.alpha <- qnorm(0.975)
  p.low.end <- pnorm(2 * z0 - z.alpha)
  p.high.end <- pnorm(2 * z0 + z.alpha)
  print(c(p.low.end, p.high.end))
  boots <- sort(boots)
  print("bias corrected C.I.")
  print(c(boots[B * p.low.end], boots[B * p.high.end]))
}
```

```
> drive.bootstrap2(aircondit, logexpmean, parboot.logexpmean, 100)
[1] "mle" "4.68290253452844"
[1] "quantiles"
      0.5%  2.5%  50%  97.5%  99.5%
3.834368 4.094229 4.710413 5.181885 5.275719
[1] "mean" "4.67889044080901"
[1] "stderr" "0.286036072515565"
[1] "bias-corrected point estimate"
[1] 4.686915
[1] "cof" "0.45"
[1] 0.01350800 0.95624129
[1] "bias corrected C.I."
[1] 3.684830 5.078217

> drive.bootstrap2(aircondit, logexpmean, parboot.logexpmean, 100)
[1] "mle" "4.68290253452844"
[1] "quantiles"
      0.5%  2.5%  50%  97.5%  99.5%
4.046429 4.096043 4.684174 5.134347 5.171321
[1] "mean" "4.66888582075081"
[1] "stderr" "0.274983035251789"
[1] "bias-corrected point estimate"
[1] 4.696919
[1] "cof" "0.5"
[1] 0.025 0.975
[1] "bias corrected C.I."
[1] 4.081475 5.125007
```

```
> drive.bootstrap2(aircondit, logexpmean, parboot.logexpmean, 1000)
[1] "mle" "4.68290253452844"
[1] "quantiles"
      0.5%  2.5%  50%  97.5%  99.5%
3.760392 3.997390 4.640964 5.166151 5.299808
```

```
[1] "mean"          "4.62143415974039"
[1] "stderr"        "0.29366260048462"
[1] "bias-corrected point estimate"
[1] 4.744371
[1] "cof"          "0.563"
[1] 0.05021169 0.98861057
[1] "bias corrected C.I."
[1] 4.113786 5.227848
```

```
> drive.bootstrap2(aircondit, logexpmean, parboot.logexpmean, 25000)
```

```
[1] "mle"          "4.68290253452844"
[1] "quantiles"
  0.5%   2.5%   50%   97.5%   99.5%
3.814595 4.033918 4.654331 5.180473 5.328618
[1] "mean"          "4.64203157705189"
[1] "stderr"        "0.292952308908569"
[1] "bias-corrected point estimate"
[1] 4.723773
[1] "cof"          "0.53676"
[1] 0.03791469 0.98400409
[1] "bias corrected C.I."
[1] 4.098616 5.225717
```

```
> drive.bootstrap2(aircondit, logexpmean, parboot.logexpmean, 25000)
```

```
[1] "mle"          "4.68290253452844"
[1] "quantiles"
  0.5%   2.5%   50%   97.5%   99.5%
3.795942 4.014882 4.654935 5.179212 5.324611
[1] "mean"          "4.64014458348023"
[1] "stderr"        "0.295191848042946"
[1] "bias-corrected point estimate"
[1] 4.725660
[1] "cof"          "0.53592"
```

```
[1] 0.03756714 0.98383410
[1] "bias corrected C.I."
[1] 4.082338 5.220944
```

## Bias-correcting bootstrap confidence intervals

- recall:

$$\begin{aligned} C\bar{D}F(q) &= Pr_*(\hat{\theta}^* \leq q) \\ &= \frac{\#\{\hat{\theta}^b \leq q\}}{B} \end{aligned}$$

- if  $C\bar{D}F(\hat{\theta}) \neq .5$ , then bias correction to percentile method c.i. may be in order
- let

$$z_0 = \Phi^{-1}(C\bar{D}F(\hat{\theta}))$$

– what Splus/R function evaluates  $\Phi^{-1}$

- then bias-corrected  $1 - \alpha$  c.i. is

$$\left[ C\bar{D}F^{-1}(\Phi(2z_0 - z_{\alpha/2})), C\bar{D}F^{-1}(\Phi(2z_0 + z_{\alpha/2})) \right]$$

– here  $z_{\alpha/2}$  is upper  $\alpha/2$  point of standard normal

$$\Phi(z_{\alpha/2}) = 1 - \alpha/2$$