

22S:166

More on the Bootstrap

Lecture 10
September 26, 2008

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

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

Using boot package for parametric bootstrap

Usage:

```
boot(data, statistic, R, sim="ordinary", stype="i",
      strata=rep(1,n), L=NULL, m=0, weights=NULL,
      ran.gen=function(d, p) d, mle=NULL, ...)
```

sim: A character string indicating the type of simulation required. Possible values are "ordinary" (the default), "parametric", "balanced", "permutation", or "antithetic". Importance resampling is specified by including importance weights; the type of importance resampling must still be specified but may only be "ordinary" or "balanced" in this case.

ran.gen: This function is used only when 'sim' is 'parametric' when it describes how random values are to be generated. It should be a function of two arguments. The first argument should be the observed data and the second argument consists of any other information needed (e.g. parameter estimates). The second argument may be a list, allowing any number of items to be passed to 'ran.gen'. The returned value should be a simulated data set of the same form as the observed data which will be passed to statistic to get a bootstrap replicate. It is important that the returned value be of the same shape and type as the original dataset. If 'ran.gen' is not specified, the default is a function which returns the original 'data' in which case all simulation should be included as part of 'statistic'. Use of 'sim="parametric"' with a suitable 'ran.gen' allows the user to implement any types of nonparametric resampling which are not supported directly.

mle: The second argument to be passed to 'ran.gen'. Typically these will be maximum likelihood estimates of the parameters. For efficiency 'mle' is often a list containing all of the objects needed by 'ran.gen' which can be calculated using the original data set only.

Example: assuming population distribution is normal

Suppose we are using the trimmed mean as a measure of center using continuous data.

```
> x <- rcauchy(25)

> trimmed.mean <- function(x) {mean(x, trim=0.25) }

ran.gen.normal <- function(d,p)
{
  rnorm( length(d), mean = p$xbar, sd = p$s)
}

boot.normal.out <- boot( data = x, statistic = trimmed.mean,
R=999, sim="parametric", ran.gen = ran.gen.normal,
mle = list( xbar = mean(x), sd = sqrt(var(x))) )

> boot.normal.out

PARAMETRIC BOOTSTRAP

Call:
boot(data = x, statistic = stat.cauchy, R = 999, sim = "parametric",
  ran.gen = ran.gen.normal, mle = list(xbar = mean(x), s = sqrt(var(x))))

Bootstrap Statistics :
  original    bias  std. error
```

```
t1* -0.1694276 0.1512959 0.7842112
> hist(boot.normal.out$t)
>
```

For Cauchy data

Since mean and variance do not exist for Cauchy distribution, choice of measures of center and spread for simulating data are somewhat arbitrary.

```
> ran.gen.cauchy <- function(d, p )
{
  rcauchy(length(d), location = p$med, scale = p$sc)
}

> boot.cauchy.out <-boot(data=x, statistic=trimmed.mean, R=999,
sim="parametric",ran.gen = ran.gen.cauchy,
mle = list( med = median(x), sc = IQR(x)/2) )
```