

# *Resampling Methods*

## *The Bootstrap*

*Javier Cabrera*

*Director, Biostatistics Institute*  
*Rutgers University*



## *The bootstrap method*

$X_1, \dots, X_N$  be a random sample from a distribution  $F$  and let  $\hat{F}_N$  be the empirical distribution of the data.

The usual statistical problem is to compute the distribution of some statistic  $T(X_1, \dots, X_N)$

$T$  could be the sample mean, the sample Variance, correlation coefficient, regression slope, ..., almost anything.

The distribution of  $T$  might be of interest for obtaining C.I. of some unknown parameter  $\theta$  or for testing some research hypothesis.

The main sticking point is that  $F$  is usually unknown. The classical solution is CLT, Maximum Likelihood, asymptotic approximations that may only work well for large samples.

## *The bootstrap method*

Bootstrap: Described by Bradley Efron (1979)

Idea:

- Replace  $F$  by  $\hat{F}_N$
- Compute the distribution of  $T(X_1, \dots, X_N)$  under  $\hat{F}_N$  and call it  $B_{T,N}$
- $B_{T,N}$  is also called the bootstrap distribution of  $T$ .

Note: Sampling from  $\hat{F}_N$  is the same as sampling from  $X_1, \dots, X_N$  with replacement.

## *Percentile, bootstrap methods*

Suppose that  $T$  is an estimator of a parameter  $\theta$ .  $\hat{\theta} = T(X_1, \dots, X_N)$

*Percentile method:*

- Compute the distribution of  $T(X_1, \dots, X_N)$  under  $\hat{F}_N$
- Calculate the percentiles of the distribution,  $(q_{0.025}, q_{0.975})$

*Bootstrap method:*

$$(2\hat{\theta} - q_{0.975}, 2\hat{\theta} - q_{0.025})$$

*is called the bootstrap method*

#### EXAMPLE 4. BOOTSTRAP

\*\*\*SAS PROGRAM\*\*\*

```
OPTIONS PS=55 LS=80;
data a;
array x {30} x1-x30;
array pp {30} pp1 -pp30;
array b {30} b1 -b30;
seed=0;
do i= 1 to 30;
    x{i} = normal(seed);
    pp{i} = 1/30;
end;
keep z ;
z = mean(of x1-x30);
do i=1 to 100;
    do j=1 to 30;
        k=round(rantbl(of seed pp1-pp30));
        b{j} = x{k};
    end;
    z = mean(of b1-b30);
    output;
end;
proc univariate normal plot;
var z;
output p1=p1 p5=p5 p10=p10 p90=p90 p99=p99;
proc print;
var p1 p5 p10 p90 p99;
run;
```

Bootstrap R package:

Other ways:

```
> x = rchisq(20,1)
```

```
> mean(x)
```

```
[1] 1.704550
```

```
> y = matrix(sample(x, 1000*20,rep=T),1000,20)
```

```
> ym = apply(y,1,mean)
```

```
> quantile(ym,c(0.025,0.975))
```

```
 2.5%  97.5%
```

```
1.100729 2.415550
```

```
> 2* mean(x) - quantile(ym,c(0.975,0.025))
```

```
 97.5%  2.5%
```

```
0.9935509 2.3083719
```

## R CODE FROM CLASS

```
x = rchisq(20,1)
plot(sort(x), (1:20)/20, type="l")
lines(seq(length=100,0,2.5), pchisq(seq(length=100,0,2.5),1))
lines(seq(length=100,0,2.5), pnorm(seq(length=100,0,2.5), mean(x), sd
(x)))
### 1000 bootstrap simulations
y = matrix(sample(x, 1000*20, rep=T), 1000, 20)
ym = apply(y, 1, mean)
hist(ym, 100)
quantile(ym, c(0.025, 0.975))
2* mean(x) - quantile(ym, c(0.975, 0.025))
mean(x) + c(-1, 1)*1.96*sd(x)
### Coefficient of Variation
cv = function(x) var(x)/mean(x)
ym = apply(y, 1, cv)
quantile(ym, c(0.025, 0.975))
### Standard deviation
ym = apply(y, 1, sd)
quantile(ym, c(0.025, 0.975))
### Bootstrap regression
stack.ls = lsfit(stack.x, stack.loss)
b = stack.ls$coef
r = stack.ls$resid
yhat = stack.loss - r
y = matrix(sample(r, 1000*21, rep=T), 1000, 21)
y = t(y) + yhat
bb = apply(y, 2, function(z, x=stack.x) lsfit(x, z)$coef)
dim(bb)
pairs(t(bb))
```