Using the boot package

Powerful, but not easy

The “boot” package is powerful, but has a learning curve!

```
require(boot)  # install 'boot' if needed
```

```
## Loading required package: boot
```

Statistic function

To use `boot()` you have have to write an auxiliary function of two inputs that computes the statistic you want to bootstrap.

```
bootMedian <- function( x, i){
  # x is a vector of data
  # i is a vector of indices for the resampled
  #   elements of x
  median(x[i]) # gets returned
}
```

Statistic function (2)

Here is a version with a print statement so we can see how it works.

```
bootMedianPrint <- function( x, i){
  # x is a vector of data
  # i is a vector of indices for the resampled
  #   elements of x
  print(i)
  print(x[i])
  median(x[i]) # gets returned
}
```

How it works

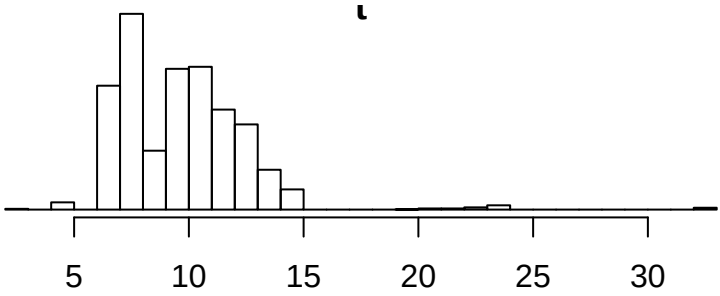
```
set.seed(1)
x <- c(.2,.3,.5,.7,1.1)
boot.tmp <- boot(x,bootMedianPrint,R=2)
```

```
## [1] 1 2 3 4 5
## [1] 0.2 0.3 0.5 0.7 1.1
## [1] 2 3 2 5 4
## [1] 0.3 0.5 0.3 1.1 0.7
## [1] 2 5 5 4 1
## [1] 0.3 1.1 1.1 0.7 0.2
```

`boot.tmp` stores the resampled statistic values and other stuff . . .

Bootstrapping the median

```
x <- c(2,7,7,8,11,13,15,33) # the original sample
set.seed(NULL); boot.object <- boot(x,bootMedian,R=5000)
par(mar=c(4.5,0,0,0))
hist(boot.object$t,breaks=40,main='t',xlab='sample median',yaxt='n')
```



Some things to try

You try:

```
# see everything saved in the results of boot
str(boot.object)
# plot a boot object
plot(boot.object)
```

Bootstrap confidence intervals

Build confidence intervals from the bootstrap object

```
boot.ci(boot.object)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 5000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.object)
```

Bootstrap confidence intervals (continued output)

```
##
## Intervals :
## Level      Normal              Basic
## 95%   ( 3.458, 14.520 )   ( 4.000, 12.000 )
##
## Level     Percentile            BCa
## 95%   ( 7, 15 )   ( 7, 14 )
## Calculations and Intervals on Original Scale
```

Bootstrap Info

- Requirements
  - sample must be representative of population
  - very small samples are still a problem
- Advantage
  - can bootstrap estimate any statistic

Bootstrap trimmed mean

```
bootTrMean <- function( x, i, trim=0){
  mean(x[i],tr=trim) # gets returned
}
set.seed(43);
x <- rlnorm(50); # very skewed distribution
boot.object <- boot(x,bootTrMean,R=5000,trim=0.2)
```

Bootstrap trimmed mean

```
boot.ci(boot.object,type='bca')
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 5000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.object, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   ( 0.823,  1.539 )
## Calculations and Intervals on Original Scale
```

Bootstrap variance

Estimate population variance (hint: it's 25)

```
bootVar <- function( x, i){
  var(x[i]) # gets returned
}
x <- rnorm(40,mean=10,sd=5)
boot.object <- boot(x,bootVar,R=5000)
```

Bootstrap variance (2)

```
boot.ci(boot.object,type='bca')
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 5000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = boot.object, type = "bca")
##
## Intervals :
## Level       BCa
## 95%   (21.65, 55.33 )
## Calculations and Intervals on Original Scale
```

Bootstrap difference using strata

First we need to set up our auxiliary stat function

```
bootMeanDiff <- function(d,i){
  # d is a dataframe, d[,1] is data, d[,2] is group factor variable
  # d[i,1] is resampled data
  print(d[i,1]) # remove or comment out for general use
  means <- tapply(d[i,1],d[,2],mean)
  means[1]-means[2]
}
```

Bootstrap difference using strata (2)

```
x <- c(1.1, 1.3, 1.4, 1.8, 2.2, 2.5, 2.6, 2.7)
group <- factor(rep(c('A','B'),c(4,4)))
d <- data.frame(x,group)
boot.object <- boot(d, bootMeanDiff, R = 5, strata = d$group )
```

```
## [1] 1.1 1.3 1.4 1.8 2.2 2.5 2.6 2.7
## [1] 1.3 1.1 1.8 1.1 2.2 2.5 2.5 2.2
## [1] 1.8 1.8 1.4 1.1 2.7 2.6 2.5 2.5
## [1] 1.1 1.4 1.3 1.4 2.7 2.6 2.5 2.7
## [1] 1.1 1.8 1.4 1.1 2.2 2.2 2.7 2.6
## [1] 1.8 1.4 1.3 1.4 2.6 2.2 2.7 2.2
```

Difference of means example

```
x <- c(rnorm(30,0,1),rnorm(30,5,4))
group <- factor(rep(c('A','B'),c(30,30)))
d <- data.frame(x,group)
boot.object <- boot(d, bootMeanDiff, R = 5000, strata = d$group)
```

Difference of means example (2)

```
boot.ci(boot.object,type='bca')$bca[4:5]
```

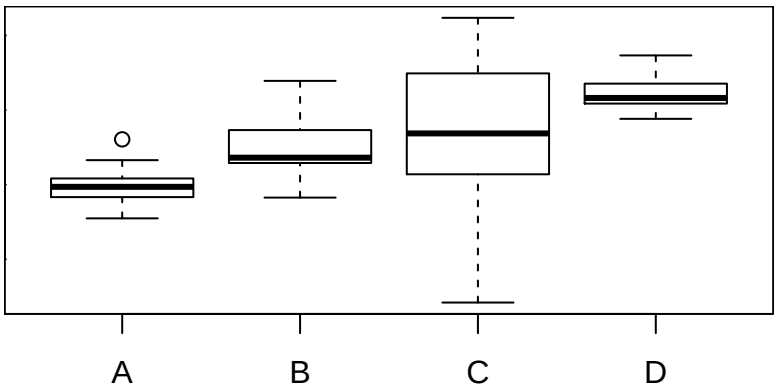
```
## [1] -6.124197 -2.898964
```

```
xA <- d$x[d$group=='A']; xB <- d$x[d$group=='B']
t.test(xA,xB)$conf.int
```

```
## [1] -6.139218 -2.759763
## attr(,"conf.level")
## [1] 0.95
```

Multiple Differences

```
x <- c(rnorm(30,0,1),rnorm(20,2,2),rnorm(40,5,4),rnorm(20,6,1))
group <- factor(rep(LETTERS[1:4],c(30,20,40,20)))
d <- data.frame(x,group)
```



Auxiliary Function

The auxiliary function can returns as many statistics as we like

```
bootMeanDiff <- function(d,i){
  means <- tapply(d[i,1],d[,2],mean)
  c( means[2]-means[1], means[3]-means[1], means[4]-means[1],
     means[3]-means[2], means[4]-means[2], means[4]-means[3] )
  }
# build the bootstrap distribution
boot.object <- boot(d, bootMeanDiff, R = 5000, strata = d$group)
```

CI's for differences (1)

```
# mu_B - mu_A (= 2)
boot.ci(boot.object,conf = 1 - .05/6, type='bca', index=1)$bca[4:5]
```

```
## [1] 1.432375 3.930528
```

```
# mu_C - mu_A (= 5)
boot.ci(boot.object,conf = 1 - .05/6, type='bca', index=2)$bca[4:5]
```

```
## [1] 1.527191 5.673106
```

```
# mu_D - mu_A (= 6)
boot.ci(boot.object,conf = 1 - .05/6, type='bca', index=3)$bca[4:5]
```

```
## [1] 5.359795 7.024595
```

CI's for differences (2)

```
# mu_C - mu_B (= 3)
boot.ci(boot.object,conf = 1 - .05/6, type='bca', index=4)$bca[4:5]
```

```
## [1] -1.232718  3.219309
```

```
# mu_D - mu_B (= 4)
boot.ci(boot.object,conf = 1 - .05/6, type='bca', index=5)$bca[4:5]
```

```
## [1] 2.149638 4.746418
```

```
# mu_D - mu_C (= 1)
boot.ci(boot.object,conf = 1 - .05/6, type='bca', index=6)$bca[4:5]
```

```
## [1] 0.5226712 4.7266846
```

CI's for differences (3)

`onewayComp()` also do pairwise Studentized CI's for differences of means

```
source('./onewayComp.R') # or in DS705data package
onewayComp(x~group,data=d,alpha=.05/6,nboot=0,
           var.equal=F, adjust='none')$comp[,2:3]
```

```
##             lwr      upr
## B-A   1.2075870 3.998288
## C-A   1.5400233 5.911862
## D-A   5.3123228 7.066007
## C-B  -1.2924456 3.538456
## D-B   2.1596852 5.012770
## D-C   0.2575984 4.668846
```

CI's for differences (4)

onewayComp() also has a bootstrap option

```
source('./onewayComp.R')
onewayComp(x~group,data=d,alpha=.05/6,nboot=10000,
           var.equal=F, adjust='none')$comp[,2:3]
```

```
##             lwr      upr
## B-A   1.3491295 4.268054
## C-A   1.3583567 5.790445
## D-A   5.3078123 7.080277
## C-B  -1.5553501 3.477992
## D-B   1.9357685 4.893643
## D-C   0.3867774 4.902089
```