Bootstrapping

```
library(boot)
library(coin)
```

Bootstrapping

Knowing the sampling distribution of a statistic tells us about statistical uncertainty (standard errors, confidence intervals). To obtain standard errors, we need sampling distributions, the distribution of sample statistics computed for different samples of the same size from the same population. A sampling distribution shows us how the sample statistic varies from sample to sample. The standard error measures how much the statistic varies from sample to sample and can be calculated as the standard deviation of the sampling distribution.

Of course, in reality we have only one sample from the population. How can we construct a sampling distribution from one sample... Theoretical statistics, namely the Central Limit Theorem, gives us the sampling distribution in some cases but what if the sampling distribution is unknown.

Bootstrapping generates an empirical distribution of a test statistic or set of test statistics by repeated random sampling with replacement from the original sample. It allows you to generate confidence intervals and test statistical hypotheses without having to assume a specific underlying theoretical distribution.

The bootstrap is a flexible and powerful statistical tool that can be used to quantify the uncertainty associated with a given estimator. For example, it can provide an estimate of the standard error of a coefficient or a confidence interval for that coefficient.

The bootstrap approach allows us to use a computer to mimic the process of obtaining additional samples, so that we can estimate the variability of our estimate. Rather than repeatedly obtaining independent samples from the population, we instead obtain distinct simulated samples by repeatedly sampling observations from the original data set with replacement. Each of these simulated samples is created by sampling with replacement, and is the same size as our original dataset. As a result some observations may appear more than once in a given bootstrap data set and some not at all.

Let's take an example of calculating the 95% confidence interval for a sample mean. The sample has 10 observations, a sample mean of 40, and a sample standard deviation of 5. Recall from introductory biostats, assuming the sampling distribution of the mean is normally distributed, the 95% CI can be calculated using:

$\bar{X} \pm t_{crit}\left(\frac{s}{\sqrt{n}}\right)$

and we would obtain 36.4232155, 43.5767845. But what if you are not willing to assume that the sampling distribution of the mean is normally distributed? You can use a bootstrapping approach instead:

1. Randomly select 10 observations from the sample, with replacement after each selection. Some observations may be selected more than once, and some may not be selected at all.

2. Calculate and record the sample mean.

3. Repeat the first two steps R, usually 1000, times.

4. Order the $R = 1000$ sample means from smallest to largest.

5. Obtain empirical CIs by determining the $(\alpha/2) \times 100\%$ and $(1 - \alpha/2) \times 100\%$ percentiles of the distribution. Lower and upper bounds of a $g = 100(1 - \alpha)\%$ CI are defined, respectively, as the $.5(1 - g/100)R$ and $1 + .5(1 + g/100)R$ values in the sorted distribution. For example, for $\alpha = .05$, $R = 1000$ bootstrap samples, and $g = 95$, the lower and upper bounds of the interval would be the 25th and the 975th values in the sorted distribution. These values are the 2.5th and 97.5th percentiles and are the 95% confidence limits.

What if you wanted confidence intervals for the sample median or the difference between two sample medians? There are no simple normal theory formulas here, and bootstrapping is the approach of choice. If the underlying distributions are unknown, if outliers are a problem, if sample sizes are small, or if parametric approaches do not exist, bootstrapping can often provide a useful method of generating confidence intervals and testing hypotheses.

Another commonly encountered example is CIs and/or SEs for the product of two regression coefficients. This is encountered when testing mediation. Although the distribution of each regression coefficient is normally distributed, the *product* of two normally distributed regression coefficients is *not*.

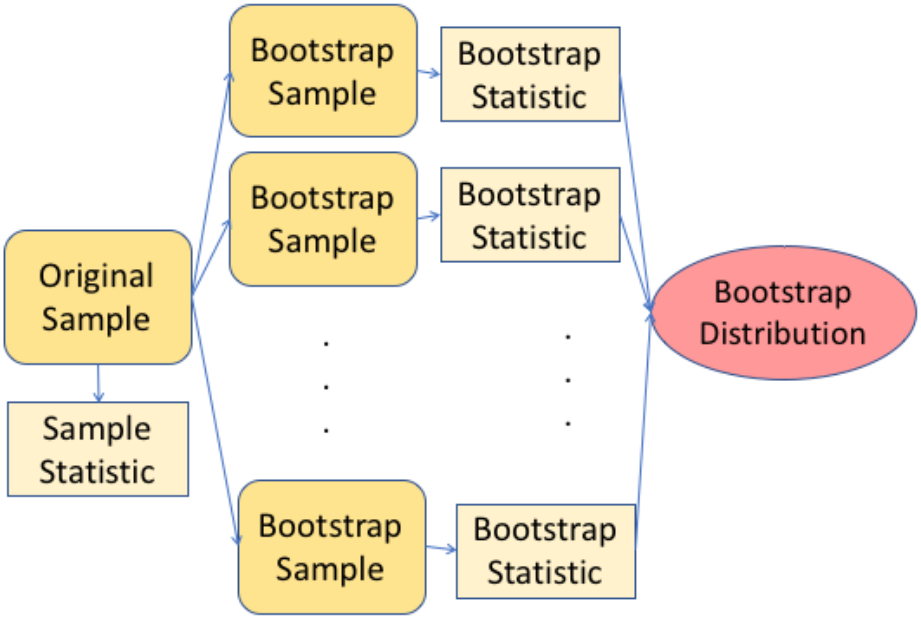


Figure 1: diagram illustrating bootstrapping

The bootstrap estimate for a parameter of interest, θ, is

$$\widehat{\theta}_{boot} = \frac{1}{R} \sum_{r=1}^{R} \widehat{\theta}^{(r)}$$

with variance estimate

$$\widehat{V}_{boot} = \frac{1}{R-1} \sum_{r=1}^{R} (\widehat{\theta}^{(r)} - \widehat{\theta}_{boot})^2$$

Sometimes we simulate from the model we are estimating (*parametric bootstrap*). Sometimes we simulate

by re-sampling the original data (*nonparametric bootstrap*), which is what we did in the above example. As always, stronger assumptions mean less uncertainty if we are right.

Increasing the number of bootstrap samples will minimize random fluctuation from simulation to simulation, but as long as it is large, this number does not matter much. Note that the number of bootstrap samples is NOT the sample size.

Bootstrapping will work for regression as well as more complicated models but figuring out the appropriate way to generate bootstrap samples can require some thought in more complex situations (e.g., multilevel data, longitudinal data).

There are different approaches to generating bootstrapped confidence intervals. In the example above, we used the *percentile method*. Another approach is the *bias-corrected and accelerated (BCa) method*, which makes simple adjustments for bias, and is preferable in most cases.

Define z_l and z_u as the corresponding z-scores in a standard normal distribution.

Define z'_l and z'_u as the z-scores defining the percentiles for the BCa bootstrap CI.

Let z_0 be the z-score corresponding to the percent of the R bootstrap estimates that are less than the estimate in the original sample. Then,

$$z'_l = z_0 + \frac{z_0 + z_l}{1 - a(z_0 + z_l)}$$

Obtain z'_u by replacing z_l with z_u in the above equation.

The acceleration constant, a, is defined as:

$$a = \frac{\sum_{1=1}^{n}(\hat{\theta} - \theta_{-i})^3}{6[\sum_{1=1}^{n}(\hat{\theta} - \theta_{-i})^2]^{3/2}}$$

where θ_{-i} is estimate of θ with observation i removed. Setting $a = 0$ results in the *bias-corrected (BC)* bootstrap CI, which does not require estimating the acceleration constant.

Continuing with the regression analysis from a few weeks ago, we are going to bootstrap the CIs. For reference, here is the regression.

```
helpdata <- read.csv("help.csv")
lm2 <- lm(i1 ~ substance + age, data=helpdata)
summary(lm2)
```

```
##
## Call:
## lm(formula = i1 ~ substance + age, data = helpdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -31.500  -9.838  -4.775   5.397 108.617
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)       20.2001     4.5075   4.481 9.42e-06 ***
## substancecocaine -16.1885     2.0108  -8.051 7.46e-15 ***
## substanceheroin  -19.1939     2.1509  -8.924  < 2e-16 ***
## age                0.2354     0.1127   2.089   0.0373 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 17.79 on 449 degrees of freedom
## Multiple R-squared:  0.2158, Adjusted R-squared:  0.2106
## F-statistic: 41.19 on 3 and 449 DF,  p-value: < 2.2e-16
```

First, we need to write a function that returns the set of regression coefficients as a vector.

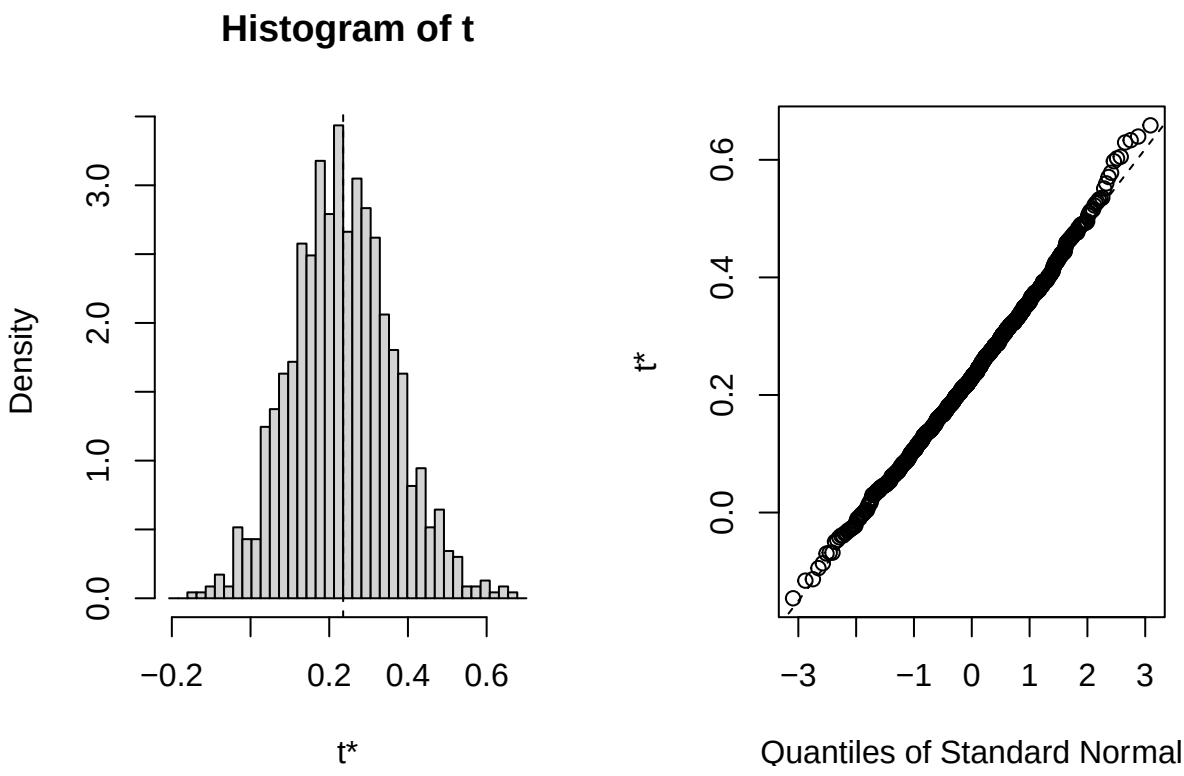
```
bs <- function(formula, data, indices) {
        d <- data[indices,] #required for boot() to be able to select samples
        fit <- lm(formula, data=d)
        return(coef(fit))
}
```

The next step is to process this function `bs` through the `boot()` function to obtain 1000 bootstrap replicates of the statistics.

```
results <- boot(data=helpdata, statistic=bs, R=1000, formula=i1 ~ substance + age)
print(results)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = helpdata, statistic = bs, R = 1000, formula = i1 ~
##     substance + age)
##
##
## Bootstrap Statistics :
##        original       bias    std. error
## t1*  20.2001176  0.117065543   4.5998273
## t2* -16.1884929 -0.074688663   1.9888123
## t3* -19.1938934 -0.116727863   1.9746754
## t4*   0.2354059 -0.001227218   0.1281024
```

```
plot(results, index=4)
```



The last step is to use the `boot.ci()` function to obtain CIs for the statistics generated in the previous step.

```
boot.ci(results, type=c("perc", "bca"), index=3)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, type = c("perc", "bca"), index = 3)
##
## Intervals :
## Level     Percentile            BCa
## 95%   (-23.14, -15.48 )   (-22.80, -15.23 )
## Calculations and Intervals on Original Scale
```

```
boot.ci(results, type=c("perc", "bca"), index=4)
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, type = c("perc", "bca"), index = 4)
##
## Intervals :
## Level     Percentile            BCa
## 95%   (-0.0102,  0.4931 )   ( 0.0235,  0.5338 )
## Calculations and Intervals on Original Scale
```

For reference, here are the CIs from the original regression:

```
confint(lm2)
```

```
##                          2.5 %      97.5 %
## (Intercept)       11.34175697  29.0584783
## substancecocaine -20.14033837 -12.2366475
## substanceheroin  -23.42096050 -14.9668264
## age                0.01393498   0.4568768
```