Bootstrapping101

Psych 5066
Washington University in St. Louis
09/27/2016

What's the difference between simulations and bootstrapping?

- **Bootstrapping:** Data —> Model

- **Simulation:** Model —> Data

Bootstrapping, conceptually

- Resampling technique with replacement
- Allows you to estimate the sampling distribution of a statistic (e.g., confident intervals, bias, variance)
- Bootstrapping is useful for making an estimation of the sampling distribution of the population
- “permutation testing” and “jackknife” techniques

R package: boot

- the “boot” package repeatedly calls your estimation function, and each time, the bootstrap sample is supplied using an integer vector of indexes

- To install & load:

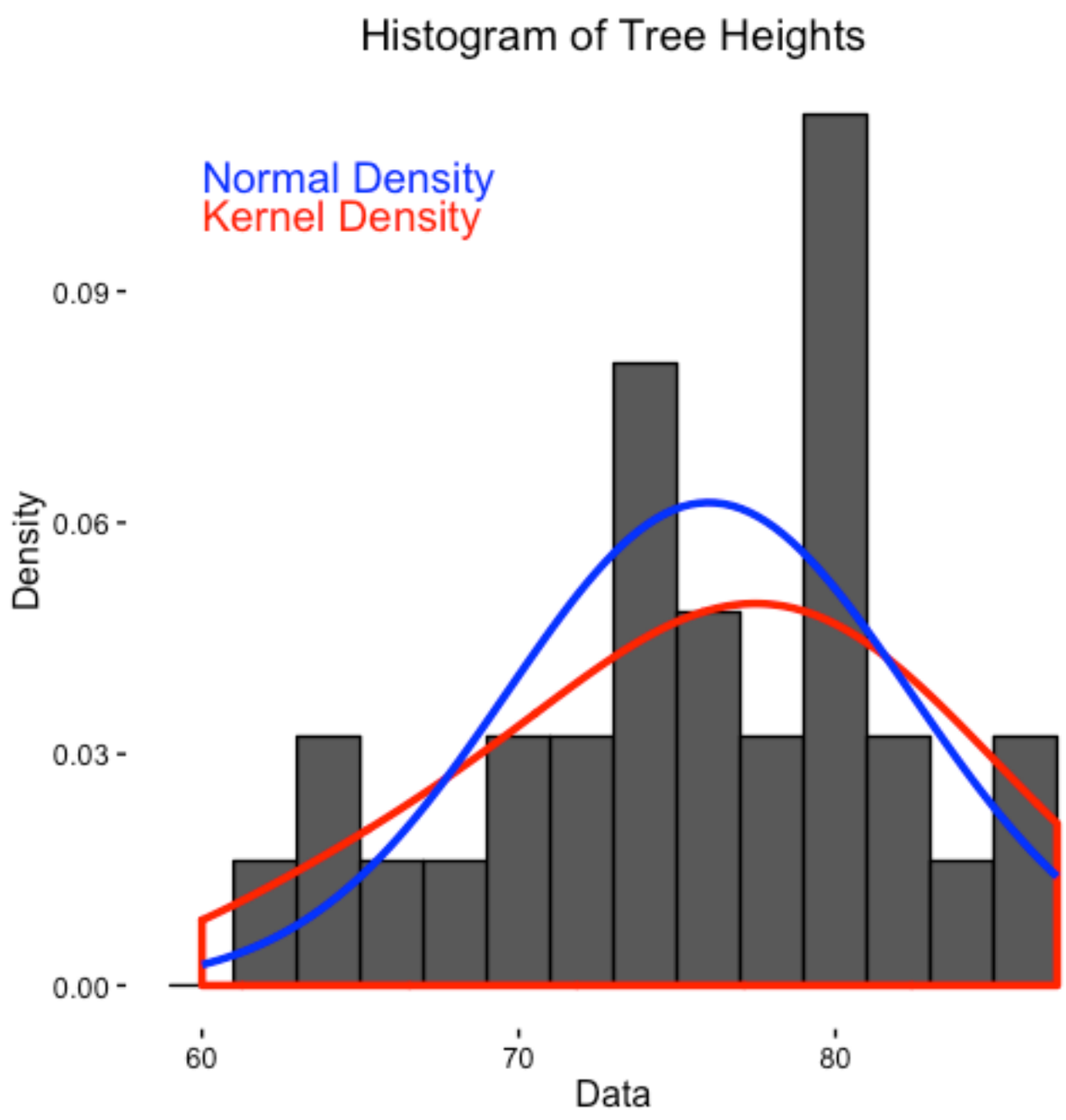
```
install.packages(“boot”)

library(boot)
```

Trees data

- We're going to work with the trees data in R.
- To look at the description of the data, type `help(trees)`
- To look at the top rows of the data, type `head(trees)`
- To look at the summary stats of the variables, type `describe(trees)`

distribution of tree heights



R package: boot

- the “boot” package repeatedly calls your ***estimation function***, and each time, the bootstrap sample is supplied using an integer vector of indexes

- To install & load:

```
install.packages(“boot”)

library(boot)

library(psych)
```

There are two steps to bootstrapping with the “boot” package

1.Create an estimation function

This is a function that you create that will return a vector containing the statistic(s) of interest.

2. Run the bootstrapping procedure

Generates replicates of a statistic applied to the data. You can will determine how many bootstraps to run (i.e., how many times to resample the statistic)

Creating Functions in R

- A great feature of R is that we can easily add functions or “verbs” to our data. They are add to the vocabulary of the R language

```
my function <- function (arg1, arg2, arg3) {
   statements of what you will do to args
   return (object)
}
```

Creating an Estimation Function

- Let's say we want to create a function that estimates the mean heights of the trees dataset

- NOTE: this particular function I've created requires the whole data frame as input. You can create similar functions that accomplish the same goal, but look different

```
trees_estimate <- function(data, indices) {
  d = data[indices,]
  mean_height = mean(d$Height)
  return (mean_height)
}
```

Let's walk through the function we just made

- In our function, we input two arguments (trees data, indices of interest) and output a mean height estimate based on the index 1 and 2 of trees data.

```
est <-trees_estimate(data = trees,indices = c(1,2))
```

- As a sanity check, we can compute this!

Using boot()

- the boot function requires at least 3 arguments

```
boot(data=trees, statistic=trees_estimate, R=100)
```

Dataset from which statistics will be calculated

Function we created to calculate statistics on each bootstrap sample

Number of bootstrap samples (i.e., iterations)

Running the simulation with different number of bootstraps

```
b1 <- boot(data=trees, statistic=trees_estimate, R=100)

b2 <- boot(data=trees, statistic=trees_estimate, R=500)

b3 <- boot(data=trees, statistic=trees_estimate, R=10000)
```

To view your output, type

```
print(b1)
print(b2)
print(b2)
```

Interpret your bootstrap

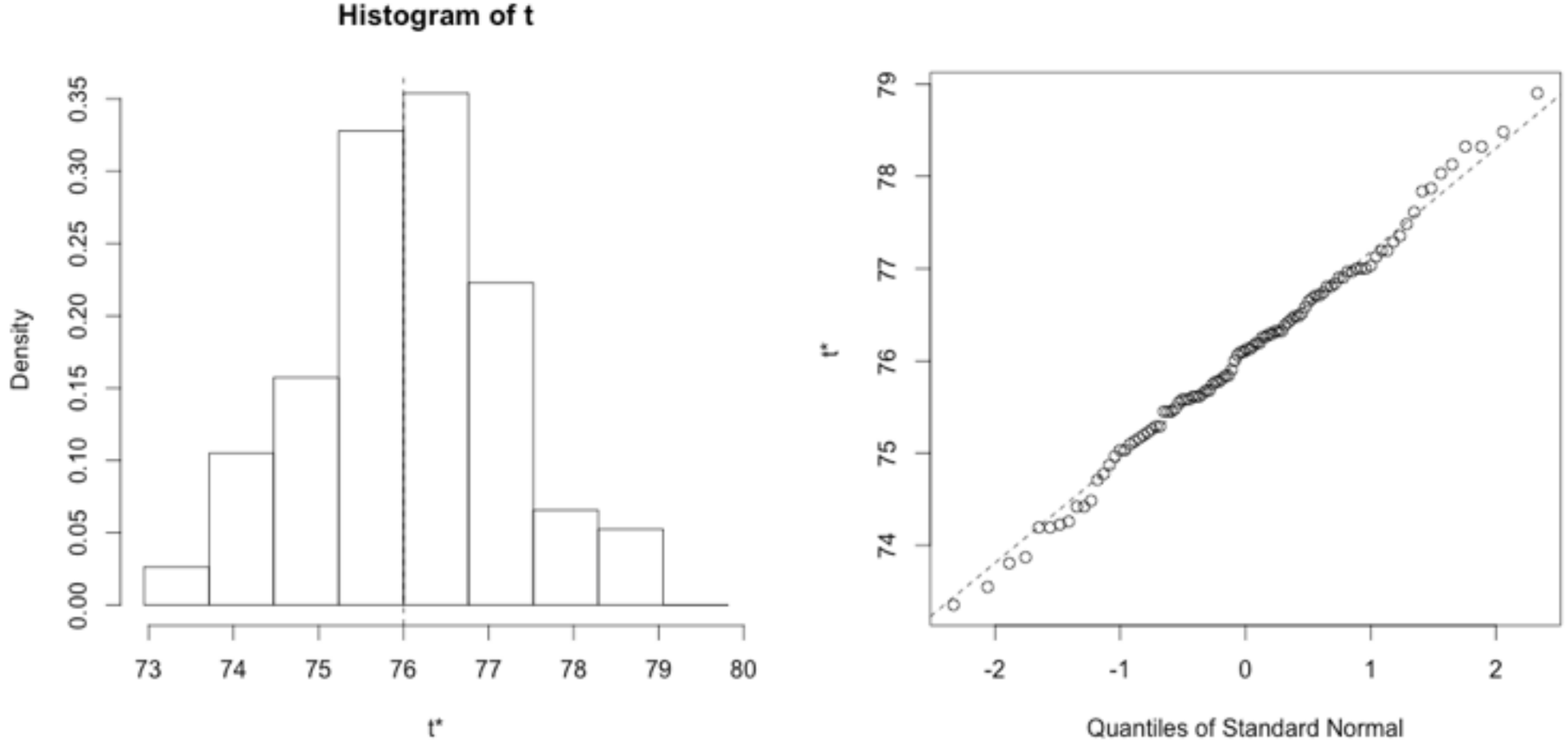
- The boot object gives you the calculated statistics:
  - t0 = the observed value of statistic applied to data
  - t = matrix with all of the estimates for each statistic
  - bias = difference between mean of sampling distribution and actual mean
  - std. error = standard deviation of sampling distribution

```
Bootstrap Statistics :
    original      bias    std. error
t1*       76 -0.04580645    1.151959
```

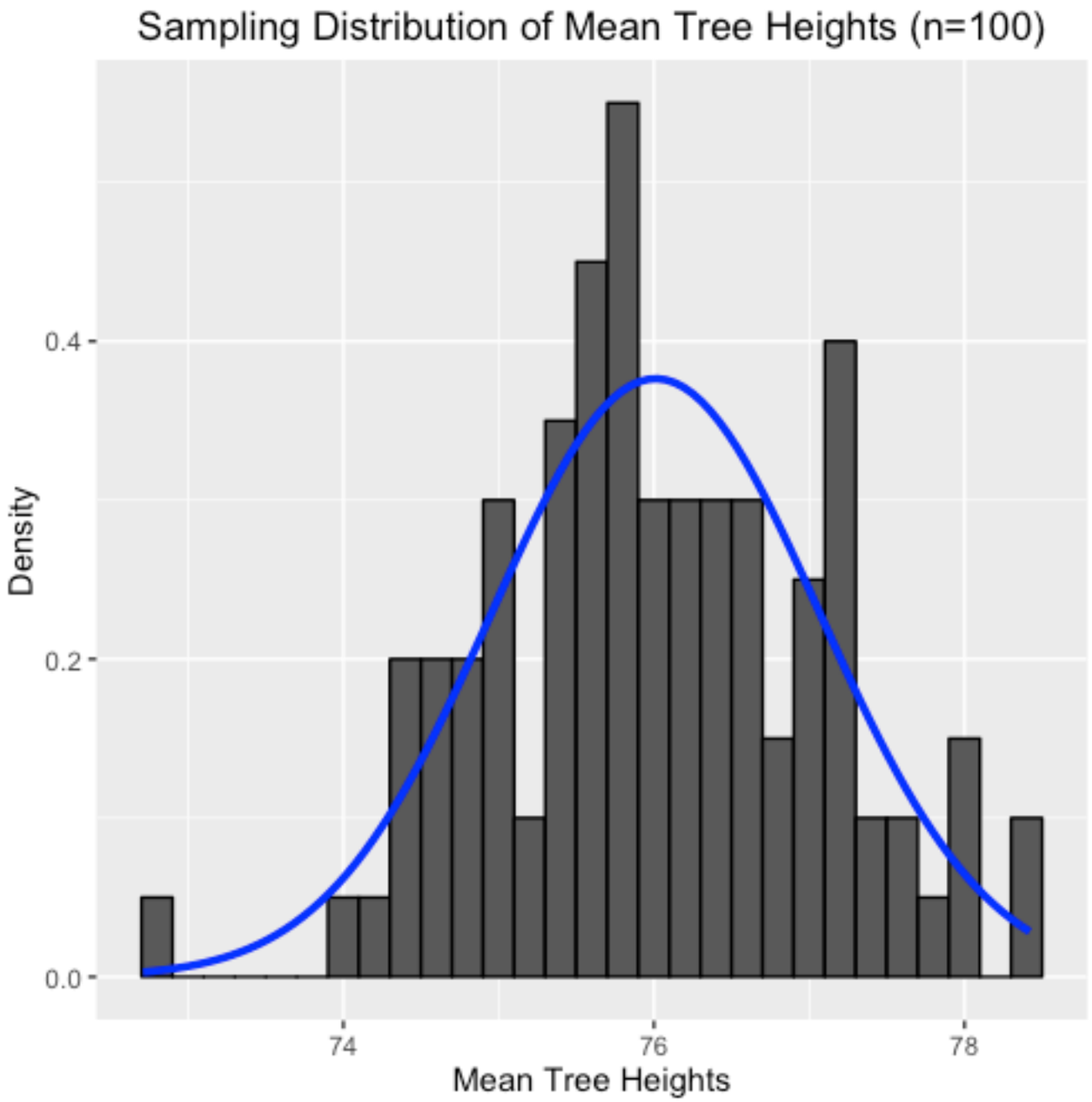
Plotting the boot objects

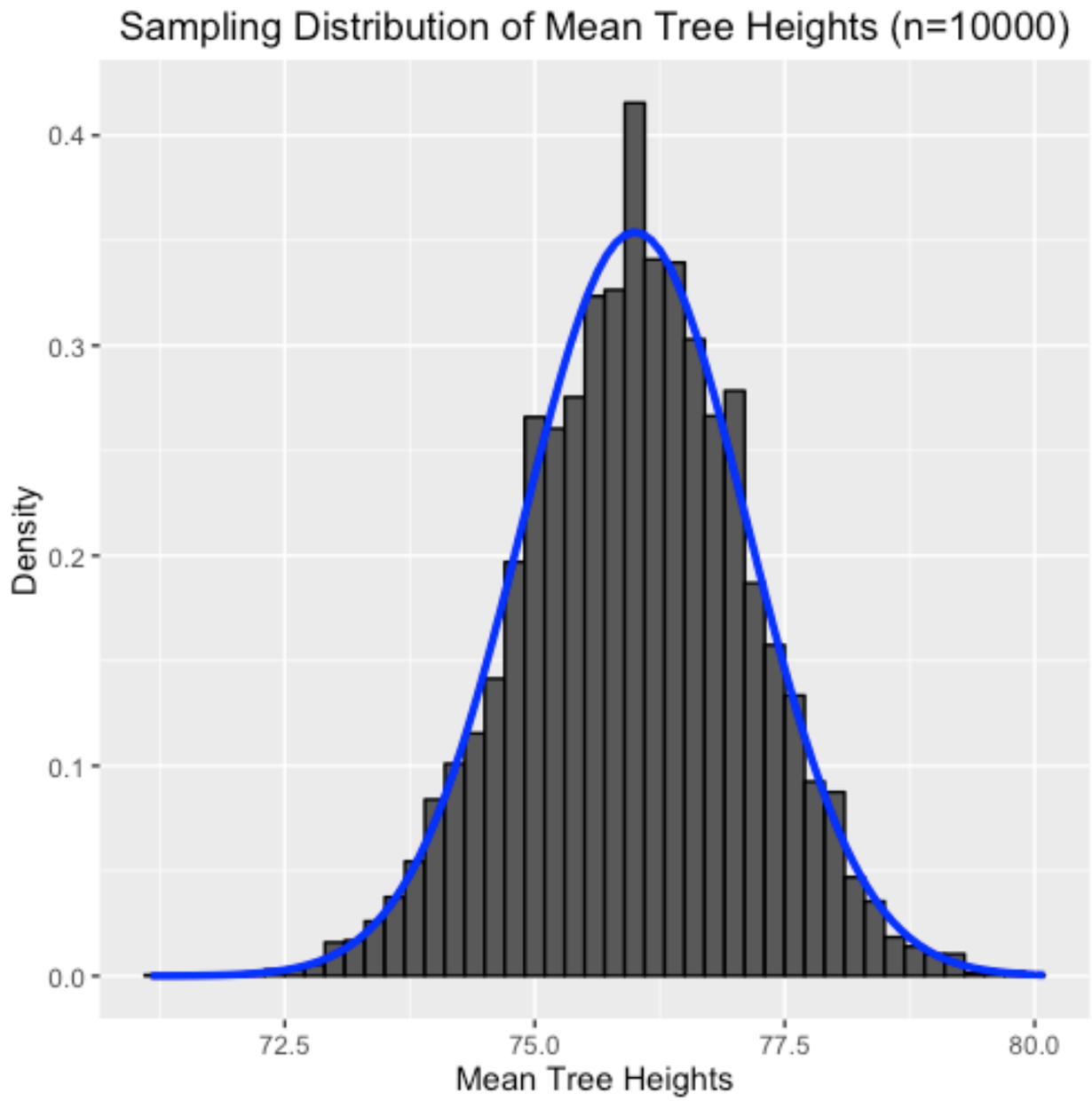
- This provides a histogram and Q-Q plot

```
plot(b1, index=1)
```



Sampling Distribution of bootstrapped mean tree heights





What if we want to estimate multiple statistics?

- You can modify your estimation function to return multiple objects

```
trees_estimate <- function(data, indices) {
  d = data[indices,]
  mean_height = mean(d$Height)
  sd_height = sd(d$Height)
  N = length(indices)
  t_height = mean_height/(sd_height/sqrt(N))
  stat = c(mean_height,sd_height,t_height)
  return (stat)
}
```

Run the bootstrap again

```
b4 <-boot(data=trees, statistic=trees_estimate_multi,
R=100)

print(b4)
```

```
Bootstrap Statistics :
     original        bias    std. error
t1* 76.000000  0.006129032   1.2488841
t2*  6.371813 -0.142528881   0.6596065
t3* 66.409685  3.02094916    8.1191390
```

(mean)

(standard dev)

(t-statistic)

NOTE: t* corresponds to the index of the “stats” vector

Calculate 95% confidence intervals for each of the bootstrapped values

- We can use boot.ci() to generate up to 5 types of equi-tailed two-sided nonparametric confidence intervals
- Here, let's create 95% CI for the tree heights we just estimated with our bootstrap, and save it to a new variable, "ci_h"

```
ci_boot <- boot.ci(boot.out=b4, conf = 0.95,
type = "perc")
```

Plotting density histogram with confidence intervals