Lab 4A: Foundations for statistical inference - Sampling distributions

In this lab, we investigate the ways in which the statistics from a random sample of data can serve as point estimates for population parameters. We're interested in formulating a *sampling distribution* of our estimate in order to learn about the properties of the estimate, such as its distribution.

The data

The data is this lab was collected by the Centers for Disease Control and Prevention (CDC) in Atlanta, GA. The Behavioral Risk Factor Surveillance System (BRFSS) is an annual telephone survey of 350,000 people in the United States. As its name implies, the BRFSS is designed to identify risk factors in the adult population and report emerging health trends. For example, respondents are asked about their diet and weekly physical activity, their HIV/AIDS status, possible tobacco use, and even their level of healthcare coverage. The BRFSS Web site (*http://www.cdc.gov/brfss*) contains a complete description of the survey, including the research questions that motivate the study and many interesting results derived from the data.

We will focus on a random sample of 20,000 people from the BRFSS survey conducted in 2000. While there are over 200 variables in this data set, we will work with a small subset.

We begin by loading the data set of 20,000 observations into R and looking at the first few observations.

```
source("http://www.openintro.org/stat/data/cdc.R")
head(cdc)
```

Let's consider the distribution of weight in this population by looking at a few summary statistics and making a histogram.

```
summary(cdc$weight)
```

```
hist(cdc$weight)
```

> **Exercise 1** Describe this distribution.

The unknown sampling distribution

Statisticians rarely have access to everyone in their target population because gathering information on an entire population is often extremely costly or impossible. Because of this, we often take a sample of the population and use that to understand the properties of the population.

If we were interested in estimating the mean weight of young adults living in the United States between the age of 18 and 30, we can do this by looking at *samples* of the population.

First, subset the data to only include individuals between the ages of 18 and 30 (young adults).

```
young <- subset(cdc, cdc$age >= 18 & cdc$age <= 30)
```

Now, to demonstrate how samples of a population can be used to estimate the true mean, we can run the following code to sample the population of young adults.

This is a product of statsTeachR that is released under a Creative Commons Attribution-ShareAlike 3.0 Unported. This lab was adapted for statsTeachR by Sara Nuñez, Nicholas Reich and Andrea Foulkes from an OpenIntro Statistics lab written by Andrew Bray and Mine Çetinkaya-Rundel.

```
samp1 <- sample(young$weight, 50)
```

This command collects a simple random sample of size 50 from the vector young$weight, which is assigned to samp1. This is like going into the BRFSS database and pulling up the files of 50 random indiviuals in this population. Working with these 50 files would be considerably simpler than working with the entire population.

Exercise 2 Describe the distribution of this sample? How does it compare to the distribution of the population from which it was sampled from? (**Note:** you can easily flip back and forth between plots by using the blue arrows in the top left corner of the plot display window)

Now, to estimate the average weight of young adults in the U.S. using the sample, our best single guess is the sample mean.

```
mean(samp1)
```

Depending on which 50 individuals you selected, your estimate could be a bit above or a bit below the true population mean of 163.50 lbs. In general, though, the sample mean turns out to be a pretty good estimate of the average weight, and we were able to get it by sampling about 1% of the population.

Exercise 3 Take a second sample, also of size 50, and call it samp2. How does the mean of samp2 compare with the mean of samp1? Suppose we took two more samples, one of size 100 and one of size 1000. Which would you think would provide a more accurate estimate of the population mean?

Not surprisingly, every time we take another random sample, we get a different sample mean. It's useful to get a sense of just how much variability we should expect when estimating the population mean this way. The distribution of sample means, called the *sampling distribution*, can help us understand this variability. In this lab, because we have access to the population, we can build up the sampling distribution for the sample mean by repeating the above steps many times. Here we will generate 5000 samples and compute the sample mean of each.

```
sample_means50 <- rep(0, 5000)

for (i in 1:5000) {
    samp <- sample(young$weight, 50)
    sample_means50[i] <- mean(samp)
}

hist(sample_means50)
```

If you would like to adjust the bin width of your histogram to show a little more detail, you can do so by changing the breaks argument.

```
hist(sample_means50, breaks = 50)
```

Here we use R to take 5000 samples of size 50 from the population, calculate the mean of each sample, and store each result in a vector called sample_means50. In the next section, we'll review how this set of code works.

Exercise 4 How many elements are there in sample_means50? Describe the sampling distribution, and be sure to specifically note its center. Would you expect the distribution to change if we instead

2

collected 50,000 sample means?

Interlude: The `for` loop

Let's take a break from the statistics for a moment to let that last block of code sink in. You have just run a for loop, a cornerstone of computer programming. The idea behind the for loop is *iteration*: it allows you to execute code as many times as you want without having to type out every iteration. In the case above, we wanted to iterate the two lines of code inside the curly braces that take a random sample of size 50 from young$weight then save the mean of that sample into the sample_means50 vector. Without the for loop, this would be painful:

```
sample_means50 <- rep(0, 5000)

samp <- sample(young$weight, 50)
sample_means50[1] <- mean(samp)

samp <- sample(young$weight, 50)
sample_means50[2] <- mean(samp)

samp <- sample(young$weight, 50)
sample_means50[3] <- mean(samp)

samp <- sample(young$weight, 50)
sample_means50[4] <- mean(samp)
```

and so on...

With the for loop, these thousands of lines of code are compressed into a handful of lines. We've added one extra line to the code below, which prints the variable i during each iteration of the for loop. Run this code.

```
sample_means50 <- rep(0, 5000)

for (i in 1:5000) {
    samp <- sample(young$weight, 50)
    sample_means50[i] <- mean(samp)
    print(i)
}
```

Let's consider this code line by line to figure out what it does. In the first line we *initialized a vector*. In this case, we created a vector of 5000 zeros called sample_means50. This vector will will store values generated within the for loop (the zeros will be replaced by values).

The second line calls the for loop itself. The syntax can be loosely read as, "for every element i from 1 to 5000, run the following lines of code". You can think of i as the counter that keeps track of which loop you're on. Therefore, more precisely, the loop will run once when i=1, then once when i=2, and so on up to i=5000.

The body of the for loop is the part inside the curly braces, and this set of code is run for each value of i. Here, on every loop, we take a random sample of size 50 from young$weight, take its mean, and store it as the ith element of sample_means50.

In order to display that this is really happening, we asked R to print i at each iteration. This line of code is optional and is only used for displaying what's going on while the for loop is running.

The for loop allows us to not just run the code 5000 times, but to neatly package the results, element by element, into the empty vector that we initialized at the outset (the vector of zeros).

Exercise 5 To make sure you understand what you've done in this loop, try running a smaller version. Initialize a vector of 100 zeros called sample_means_small. Run a loop that takes a sample of size 50 from young$weight and stores the sample mean in sample_means_small, but only iterate from 1 to 100. Print the output to your screen (type sample_means_small into the console and press enter). How many elements are there in this object called sample_means_small? What does each element represent?

Sample size and the sampling distribution

Mechanics aside, let's return to the reason we used a for loop: to compute a sampling distribution, specifically, this one.

```
hist(sample_means50, breaks = 25)
```

The sampling distribution that we computed tells us a lot about estimating the average weight of young adults living in the United States in the year 2000. Because the sample mean is an unbiased estimator, the sampling distribution is centered at the true average weight of the given population, and the spread of the distribution indicates how much variability is induced by sampling only 50 weights.

To get a sense of the effect that sample size has on our distribution, let's build up two more sampling distributions: one based on a sample size of 10 and another based on a sample size of 100.

```
sample_means10 <- rep(0, 5000)
sample_means100 <- rep(0, 5000)

for (i in 1:5000) {
    samp <- sample(young$weight, 10)
    sample_means10[i] <- mean(samp)
    samp <- sample(young$weight, 100)
    sample_means100[i] <- mean(samp)
}
```

Here we're able to use a single for loop to build two distributions by adding additional lines inside the curly braces. Don't worry about the fact that samp is used for the name of two different objects. In the second command of the for loop, the mean of samp is saved to the relevant place in the vector sample_means10. With the mean saved, we're now free to overwrite the object samp with a new sample, this time of size 100. In general, anytime you create an object using a name that is already in use, the old object will get replaced with the new one.

To see the effect that different sample sizes have on the sampling distribution, plot the three distributions on top of one another.

```
par(mfrow = c(3, 1))

xlimits = range(sample_means10)

hist(sample_means10, breaks = 20, xlim = xlimits)
hist(sample_means50, breaks = 20, xlim = xlimits)
hist(sample_means100, breaks = 20, xlim = xlimits)
```

The first command specifies that you'd like to divide the plotting area into 3 rows and 1 column of plots. (**Note:** you may need to zoom in on your plot to see the three histograms.)

To return to the default setting of plotting one plot at a time, run the following command:

```
par(mfrow = c(1, 1))
```

The breaks argument specifies the number of bins used in constructing the histogram. The xlim argument specifies the range of the x-axis of the histogram, and by setting it equal to xlimits for each histogram, we ensure that all three histograms will be plotted with the same limits on the x-axis.

Exercise 6 When the sample size is larger, what happens to the center? What about the spread?

On your own

So far, we have only focused on estimating the mean weight of young adults living in the United States in 2000. Now you'll try to estimate the mean *desired* weight of the same population (variable *wtdesire*).

1. Take a random sample of size 50 from young$wtdesire. Using this sample, what is your best point estimate of the population mean?

2. Since you have access to the population, simulate the sampling distribution for $\bar{x}_{young\$wtdesire}$ by taking 5000 samples from the population of size 50 and computing 5000 sample means. Store these means in a vector called sample_means50. Plot the data, then describe the shape of this sampling distribution. Based on this sampling distribution, what would you guess the mean desired weight of this population is? Finally, calculate and report the population mean.

3. Change your sample size from 50 to 300, then compute the sampling distribution using the same method as above, and store these means in a new vector called sample_means300. Describe the shape of this sampling distribution, and compare it to the sampling distribution for a sample size of 50. Based on this sampling distribution, what would you guess to be the mean desired weight of individuals in this population?

4. Of the sampling distributions from 2 and 3, which has a smaller spread? If we're concerned with making estimates that are more often close to the true value, would we prefer a distribution with a large or small spread?

5. Create a new variable wdiff that calculates the difference between actual weight and desired weight (weight - wtdesire). Repeat questions 2 and 3 for this new variable to estimate the mean difference in weight and desired weight in the young adult population. What can you conclude about how this population feels about their weight (on average)?