Problem Set 05: Sampling

Professor Bradley Warner

June, 2022

Documentation: None

Introduction

For this problem set, we will mimic the tactile sampling we read about with virtual sampling. We will use some data from the general social survey, an annual personal-interview survey conducted in the United States. The survey is designed to monitor changes in both social characteristics and attitudes.

For this problem set, the **population** of interest will be **ALL** 2538 individuals living in a single neighborhood in 2014. As an analogy to the tactile sampling you did in class, the neighborhood is the "bowl" and the 2,538 people are the little balls.

If you get stuck as you are working through this problem set, it will likely be helpful to review Chapter 7 in ModernDive, in particular subsections 7.3.1 on "Terminology & notation" and 7.3.2 on "Statistical definitions". The terminology, notation, and definitions related to sampling are definitely tricky at first; the best method to master them is practice, practice, practice.

Key Symbols:

Symbol	Population Parameter	Point Estimate
Number of cases	N	n
Proportion	p	$\hat{p}$
Standard error	SE	$\widehat{SE}$

Setup

First load the necessary packages:

```
library(ggplot2)
library(dplyr)
library(forcats)
library(moderndive)
```

The GSS data we will be working with is in the `gss_cat` data frame, which comes built-in with the `forcats` package you just loaded. You can take a `glimpse()` of the `gss_cat` data set like so:

```
data(gss_cat)
glimpse(gss_cat)
```

Problem description

In this section we will put all the pieces in place for the problem set.

Wrangling the data

This data set includes many years of data, and many variables. To start, we will restrict our analysis to only 2014, and to only the variable indicating the `marital` status of each respondent.

```
gss_14 <- gss_cat %>%
  filter(year == 2014) %>%
  select(marital)
```

The following shows all the unique responses for `marital` status:

```
gss_14  %>%
  distinct(marital)
```

```
## # A tibble: 6 x 1
##   marital
##   <fct>
## 1 Divorced
## 2 Married
## 3 Never married
## 4 Separated
## 5 Widowed
## 6 No answer
```

Setting a seed for your computer's Random Number Generator

In this problem set, will take some random samples of data using R. In order to make sure R takes the same random sample every time you run your code (so you can reproduce your work), you can do what is called "setting a seed". Do this in any code chunk where you take a random sample! Otherwise, the answers you write down might accidentally become out of sync with the output of your code when your knit your document!

You can control your computer's random number generator by providing a number to using the `set.seed` function. Any number will do - in the example below, we use 45 as our seed value.

```
set.seed(45)
```

The true population proportion p of divorced people

Again, for this problem set, the **population** of interest will be **ALL** 2,538 individuals living in this single neighborhood in 2014. Since we have data on **ALL** 2538 people living in the neighborhood, we can compute the **exact population proportion p of divorced people directly** using **ALL** the data:

```
gss_14 %>%
  summarize(divorced = sum(marital == "Divorced"),
            N = n()) %>%
  mutate(p = divorced / N)
```

```
## # A tibble: 1 x 3
##   divorced     N     p
##      <int> <int> <dbl>
## 1      411  2538 0.162
```

> Note that we used N (the size of the full population, 2,538 people) and computed p (not $\hat{p}$). And, no inference from sample to the population is needed. This is because we're working with the **entire population** of interest. We do not need to *estimate* the true proportion, or infer something about the true population proportion of divorced people in this neighborhood in 2014, because in this case, we can compute it directly (just like counting all red balls in the bowl). Thus, we know that p is exactly 0.16. In other words, this situation is not a realistic reflection of a real life problem.

For the rest of this problem set, we will be *simulating* the act of sampling from this neighborhood population to understand and study how factors like sample size influence **sampling variation**.

Demo: Sampling 50 people in the neighborhood

Estimating $\hat{p}$ from a single sample

We are first going to use random sampling to **ESTIMATE** the true **population** proportion p of the neighborhood that are divorced with only a **sample** of 50 people.

> This will represent a situation of only having the resources to knock on 50 doors to get responses from people in this neighborhood!

```
set.seed(45)

n50_1rep <- gss_14 %>%
  rep_sample_n(size = 50, reps = 1)
```

Be sure to look at the results in the viewer. And remember, you can set the seed to whatever value you like.

Next, let's calculate the **sample proportion** $\hat{p}$ of people who identified as `Divorced` in our sample of 50 people.

```
n50_1rep %>%
  summarize(divorce_count = sum(marital == "Divorced"),
            n = n()) %>%
  mutate(p_hat = divorce_count/ n)
```

```
## # A tibble: 1 x 4
##   replicate divorce_count     n p_hat
##       <int>         <int> <int> <dbl>
## 1         1             6    50  0.12
```

This sample proportion $\hat{p}$ is an **ESTIMATE**; it's our **best guess** of what the **true population** proportion p of `Divorced` people is in this neighborhood, based on a sample of only 50 people. It is reasonably close to the true population proportion $p = 0.16$ we calculated from the full population.

Exercise 1

Modify the code below to take 3 samples of 50 people instead of just 1 sample, and then compute an estimate of the proportion of Divorced people in the entire population from each sample individually. Use the seed we have provided.

```
set.seed(18)

n50_3rep <- gss_14 %>%
  rep_sample_n(size = 50, reps = 1)
```

Answer:

```
set.seed(18)

n50_3rep <- gss_14 %>%
  rep_sample_n(size = 50, reps = 3)

n50_3rep %>%
  summarize(divorce_count = sum(marital == "Divorced"),
            n = n()) %>%
  mutate(p_hat = divorce_count/ n)
```

```
## # A tibble: 3 x 4
##   replicate divorce_count     n p_hat
##       <int>         <int> <int> <dbl>
## 1         1             6    50  0.12
## 2         2             9    50  0.18
## 3         3             8    50  0.16
```

Estimating $\widehat{SE}$ from a single sample

Typically we only have the opportunity to collect **one sample** for our study, and so we have to use the amount of variability in our **single sample** as an estimate of the amount of variability we might expect in our results if we had taken a random sample of 50 different people. The $\widehat{SE}_{\hat{p}}$ serves as an **ESTIMATE** of **sampling variability** if you only have a **single sample**. The formula for estimating the standard error of $\hat{p}$ is the following:

$$\widehat{SE}_{\hat{p}} \approx \sqrt{\frac{\hat{p} \times (1 - \hat{p})}{n}}$$

> Note that we use n to represent the size of the sample and we that use $\hat{p}$ to represent the proportion of divorced people because we are ESTIMATING a proportion based on only a sample. Likewise, the SE "wears a hat" because we are ESTIMATING the true standard error based on a sample.

The standard error of $\hat{p}$ can be estimated in R like so:

```
n50_1rep %>%
  summarize(divorce_count = sum(marital == "Divorced"),
            n = n()) %>%
  mutate(p_hat = divorce_count/ n,
         se_hat = sqrt(p_hat * (1 - p_hat) / n))
```

```
## # A tibble: 1 x 5
##   replicate divorce_count     n p_hat se_hat
##       <int>         <int> <int> <dbl>  <dbl>
## 1         1             6    50  0.12 0.0460
```

Demo: Generating a sampling distribution of $\hat{p}$

If you ran the code chunk that takes a random sample of 50 people a thousand more times, and wrote down every $\hat{p}$ you got, you would have constructed a "sampling distribution" of the proportion of divorced people.

> A sampling distribution shows every (or nearly every!) possible value a point estimate can take on, along with how likely each value is to be observed, for samples **of a given size** from a population.

Sampling distribution of $\hat{p}$ for n = 50

Instead of running the sampling code chunk for n = 50 over and over, we can "collect" 1000 samples of n = 50 really easily in R. The following code chunk takes 1000 **different** samples of n = 50 and stores them in the data frame `n50_1000rep`:

```
set.seed(19)

n50_1000rep <- gss_14 %>%
  rep_sample_n(size = 50, reps = 1000)
```

Be sure to look at `n50_rep1000` in the data viewer to get a sense of these 1000 samples look like.

Exercise 2

What is the name of the column in the `n50_1000rep` data frame that identifies which of the 1000 samples an observation belongs to?

Answer: replicate

The following code chunk calculates the sample proportion $\hat{p}$ of people who reported they were divorced for each of the **1000 samples**

```
p_hat_n50_1000rep <- n50_1000rep %>%
  group_by(replicate) %>%
  summarize(divorce_count = sum(marital == "Divorced"),
            n = n()) %>%
  mutate(p_hat = divorce_count / n)
```

Take a look at the first five rows of the results:

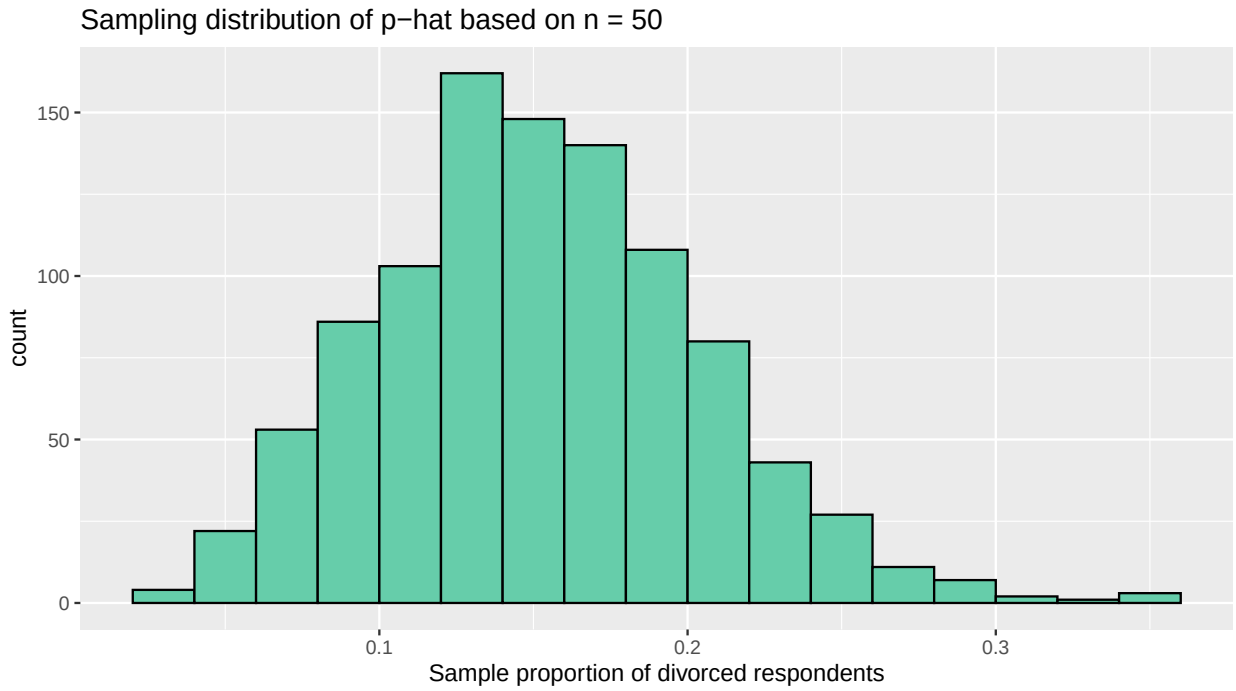
```
p_hat_n50_1000rep %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   replicate divorce_count     n p_hat
##       <int>         <int> <int> <dbl>
## 1         1             4    50  0.08
## 2         2             5    50  0.1
## 3         3             8    50  0.16
## 4         4             9    50  0.18
## 5         5            10    50  0.2
```

Visualizing the sampling distribution of $\hat{p}$ for n = 50

We can plot the **sampling distribution** of these 1000 $\hat{p}$ estimates of divorced respondents with a histogram, like so:

```
ggplot(p_hat_n50_1000rep, aes(x = p_hat)) +
  geom_histogram(binwidth = 0.02, color = "black", fill = "aquamarine3", boundary=0) +
  labs(x = "Sample proportion of divorced respondents",
       title = "Sampling distribution of p-hat based on n = 50")
```



Mean and standard error of the sampling distribution of $\hat{p}$ for n = 50

Finally we can estimate the mean of the sampling distribution by calculating the mean of all 1000 $\hat{p}$ estimates, and the standard error of the sampling distribution by calculating the standard deviation of all 1000 $\hat{p}$ values like so:

```
p_hat_n50_1000rep %>%
  summarize(M_p_hat = mean(p_hat),
            SE_p_hat = sd(p_hat))
```

```
## # A tibble: 1 x 2
##   M_p_hat SE_p_hat
##     <dbl>    <dbl>
## 1   0.162   0.0522
```

Basically, we treat the 1000 point estimates of the population proportion just like any other sample of numbers.

Exercise 3

How do the population proportion and standard error estimates computed by taking the mean and standard deviation of the 1000 simulated sample proportions compare to the estimates of $\hat{p}$ and $\widehat{SE}$ based on your **single** sample of 50 people earlier in this Problem Set?

Answer: They are very similar

Exercise 4

Use the `rep_sample_n` function to collect 1000 virtual samples of size $n = 25$. **BE SURE TO NAME YOUR SAMPLE SOMETHING NEW, TO ENSURE YOU CAN DISTINGUISH IT FROM THE n = 50 SAMPLE ABOVE!** Use a seed of 910. Then calculate the sample proportion $\hat{p}$ of people who reported they were `Divorced` for each replicate of your n = 25 sampling.

Answer:

```
set.seed(910)

n25_1000rep <- gss_14 %>%
  rep_sample_n(size = 25, reps = 1000)
```

```
n25_1000rep %>%
  group_by(replicate) %>%
  summarize(divorce_count = sum(marital == "Divorced"),
            n = n()) %>%
  mutate(p_hat = divorce_count / n)
```

```
## # A tibble: 1,000 x 4
##    replicate divorce_count     n p_hat
##        <int>         <int> <int> <dbl>
##  1         1             3    25  0.12
##  2         2             4    25  0.16
##  3         3             4    25  0.16
##  4         4             1    25  0.04
##  5         5             4    25  0.16
##  6         6             5    25  0.2
##  7         7             2    25  0.08
```

```
##  8         8             4    25  0.16
##  9         9             3    25  0.12
## 10        10             1    25  0.04
## # ... with 990 more rows
```

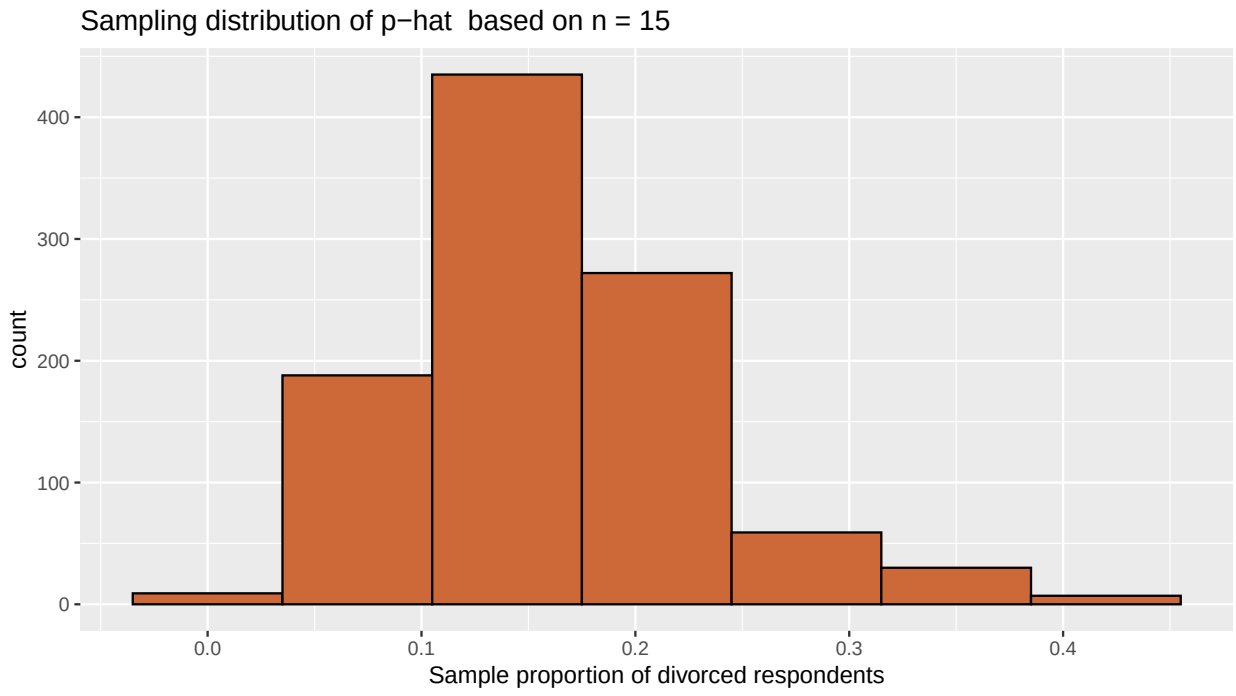
Exercise 5

Visualize the sampling distribution of $\hat{p}$ from your n = 15 sampling with a histogram

Answer:

```
p_hat_n25_1000rep <- n25_1000rep %>%
  group_by(replicate) %>%
  summarize(divorce_count = sum(marital == "Divorced"),
            n = n()) %>%
  mutate(p_hat = divorce_count / n)
```

```
ggplot(p_hat_n25_1000rep, aes(x = p_hat)) +
  geom_histogram(binwidth = 0.07, color = "black", fill = "sienna3") +
  labs(x = "Sample proportion of divorced respondents",
       title = "Sampling distribution of p-hat  based on n = 15")
```



Exercise 6

Calculate the mean of the n = 15 sampling distribution, and the standard error of the n = 15 sampling distribution

Answer:

```
p_hat_n25_1000rep %>%
  summarize(M = mean(p_hat),
            SE = sd(p_hat))
```

```
## # A tibble: 1 x 2
##       M     SE
##   <dbl>  <dbl>
## 1 0.162 0.0723
```

Exercise 7

How does the standard error of the n= 25 sampling distribution compare to the standard error of the n = 50 sampling distribution?

Answer: The standard error is much larger at n = 25. It is nearly has nearly one and a half the value of the SE of the n= 50 sampling distribution.

Exercise 8

Use the `rep_sample_n` function to collect 1000 virtual samples of size $n = 200$. **Note: BE SURE TO NAME YOUR SAMPLE SOMETHING NEW, TO ENSURE YOU CAN DISTINGUISH IT FROM THE n = 50, and n = 25 SAMPLES ABOVE!** Use a seed of 84. Calculate the proportion $\hat{p}$ of people who reported they were `Divorced`for each replicate of your n = 200 sampling.

Answer:

```
set.seed(84)

n200_1000rep <- gss_14 %>%
  rep_sample_n(size = 200, reps = 1000)
```

```
(p_hat_n200_1000rep <- n200_1000rep %>%
  group_by(replicate) %>%
  summarize(n = n(),
            divorce_count = sum(marital == "Divorced")) %>%
  mutate(p_hat = divorce_count / n))
```

```
## # A tibble: 1,000 x 4
##    replicate     n divorce_count p_hat
##        <int> <int>         <int> <dbl>
##  1         1   200            28 0.14
##  2         2   200            39 0.195
##  3         3   200            38 0.19
##  4         4   200            36 0.18
##  5         5   200            41 0.205
##  6         6   200            24 0.12
##  7         7   200            35 0.175
##  8         8   200            28 0.14
##  9         9   200            29 0.145
## 10        10   200            36 0.18
## # ... with 990 more rows
```

Exercise 9

Was there more **variability** from sample to sample when we took a sample size of 200 or when we took a sample size of 50? **Explain what evidence you have for assessing this**

Answer:

```
p_hat_n200_1000rep %>%
  summarize(M = mean(p_hat),
            SE = sd(p_hat))
```

```
## # A tibble: 1 x 2
##       M     SE
##   <dbl>  <dbl>
## 1 0.162 0.0254
```

Answer: the standard error of the sampling distribution for n = 200 was much smaller than the standard error of the sampling distribution for n = 50. . . .indicating that we had more variability from sample to sample when only collecting 50 samples.

Exercise 10

Which sampling distribution looked more normally distributed (bell shaped and symmetrical); the one built on n = 25, 50 or 200? **Why?**

Answer: The sampling distribution for n = 200 looked more normally distributed. This is a key tenet of the central limit theorem: as your sample size increases, your sampling distribution becomes more normally distributed, and narrower (i.e. smaller spread) though the n = 50 one also looked pretty bell-shaped and symmetrical

---

Documenting software

- File creation date: 2022-06-13
- R version 4.1.3 (2022-03-10)
- `tidyverse` package version: 1.3.1
- `moderndive` package version: 0.5.4
- `infer` package version: 1.0.0
- `forcats` package version: 0.5.1