Stat 145, Tue 21-Sep-2021 -- Tue 21-Sep-2021
Biostatistics
Spring 2021 $=210\text{mm}$ $=297\text{mm}$

Tuesday, September 21st 2021

Due:: WW Descriptive2 at 11 pm

Tuesday, September 21st 2021

Wk 4, Tu
Topic:: Sampling distributions 2
Read:: Lock5 3.1
HW:: PS05 due Sat.

```
die <- c(1:6)
sample(die, size=10)   # doesn't work
sample(die, size=10, replace=TRUE)    # allows repetition
  like the difference between drawing from a bag with/without replacement

sample(mlb18$HR, size=4)
with(mlb18, sample(HR, size=4))    # does the same thing

gf_density(~HR, data=mlb18) %>% gf_labs(title = "Population distribution")
gf_density(~HR, data=mlb18) %>% gf_refine(scale_x_continuous(limits=c(0,70)))

p1 <- ...
p2 <- ...
grid.arrange(p1, p2, nrow=2)
```

Class Activity: Sampling Distributions

In **statistical inference**, we consistently use a computed statistic from a sample to draw a conclusion about the population. To get that sample statistic, we first draw a random sample of some size n. But, this would get us nowhere without knowing something about the **sampling distribution** of that statistic—the values it takes, and how often they occur.

In this activity we develop some intuition about **sampling distributions for sample means**. The specific context in which this statistic arises is as follows:

- your population is a quantitative variable,
- you have drawn a random sample of size n from that population and used it to compute a sample mean $\bar{x}$.

To see what values the sample mean can take, we will operate under the usually-unrealistic conditions that you can return to the population and repeatedly draw new samples of size n, each time recalculating $\bar{x}$.

1. Let us take as the population the players of Major League Baseball who, during the 2018 season, had at least 100 at-bats. You can read this data set into RStudio with the command:

   ```
   filter(read.csv("https://www.openintro.org/data/csv/mlb_players_18.csv"), AB >= 100)
   ```

   You may decide what name to store this data as, but in what follows it will be referred to as `mlb18`.

   How would you describe the distribution of homeruns? What is the population mean μ and the population standard deviation σ?

2. **[This step guided by the professor.]** For sample size $n = 4$, do the following:

   (a) Draw one random sample of size n and from it compute the (sample) mean $\bar{x}$.

   (b) Use the `do()` command from the `mosaic` package to calculate the sample means from 5000 different samples of size n. Store the result as `manyMeansSS4`.

   (c) Make a density histogram (`gf_dhistogram()`) or density plot (`gf_density()`) of these sample means $\bar{x}$. This serves as a reasonable approximation to the **sampling distribution of $\bar{x}$ for samples of size n drawn from this population**. It should be different from the population distribution you described in Number 1. Discuss with your partner the differences you see.

   (d) Find the mean and standard deviation of the sampling distribution in part (c). These are, respectively, approximations of the mean and standard deviation of the sampling distribution of $\bar{x}$. As you acquire more experience, it will lead you to expect this mean to be fairly similar to μ (the population mean of Step 1), but that the standard deviation is noticeably different than σ. As this standard deviation came from a sampling distribution, we give it a different name, calling it the **standard error** of the sample statistic $\bar{x}$, or $\text{SE}_{\bar{x}}$.

(e) If you display your sampling distribution of $\bar{x}$ using `gf_dhistogram()` (usage is like that of `gf_histogram()`), and then follow it with the pipe and extra command as displayed

```
gf_dhistogram( ... ) %>% gf_dist("norm", mean=you supply, sd=you supply)
```

does it appear the sampling distribution is well-approximated by this normal distribution?

3. Repeat all the sub-steps of Step 2, adapting them as necessary to produce the sampling distribution of $\bar{x}$ for samples of size $n = 20$ drawn from this population. In what ways is this sampling distribution different from the previous one?

4. Again repeat the sub-steps of Step 2, this time simulating the sampling distribution of $\bar{x}$ for samples of size $n = 180$ drawn from this population. In what ways is this sampling distribution different from the previous one?

   You will likely have used the command

   ```
   sample(HR, size=180)
   ```

   in your work here, which takes an SRS of size 180 from the population containing only 436 baseball players in total. Take yet another look at the sampling distribution of $\bar{x}$ for $n = 180$, but this time adding the `replace` "switch"

   ```
   sample(HR, size=180, replace=TRUE)
   ```

   With the added switch, the resulting samples are no longer SRS, but are called i.i.d. random samples. Do you see any difference in the shape of the sampling distribution or in the standard error?

5. The "sampling distribution for $\bar{x}$" app at the website `https://shiny.calvin.edu:3838/scofield/cltMeans/` eliminates the need for you to type commands to generate the sampling distributions of $\bar{x}$. Play with the app, tweaking the underlying population and the sample size. Is there a minimal sample size n that always seems to result in a normal-looking sampling distribution?

Conclusions

Complete these sentences in a manner consistent with your observations.

1. For any set quantitative population variable, there is a different sampling distribution for the sample mean $\bar{x}$ for each fixed sample size n. As n increases, these sampling distributions change predictably . . . [you describe]
2. As n increases, the standard error $SE_{\bar{x}}$ . . .
3. When the sample size n corresponds to a relatively large portion of the overall population size, there can be a noticeable, if not always overly large, difference between sampling distributions for $\bar{x}$ obtained via SRS samples and those obtained as i.i.d. samples. The one with the larger standard error is . . .

````
---
title: "Sampling Distribution Activity"
author: "Thomas Scofield"
date: ''`r format(Sys.Date(), "%B %d, %Y")`''
output:
  pdf_document:
    fig_height: 2.2
    fig_width: 4
  html_document:
    fig_height: 2.2
    fig_width: 4
  word_document:
    fig_height: 2.2
    fig_width: 4
---

```{r, setup, include = FALSE, message=FALSE}
# load packages that are going to be used
library(mosaic)      # this loads ggformula (for plotting), etc. too
library(openintro)   # this loads data sets intro to modern stats
library(oibiostat)   # this loads data sets open intro biostats
library(pander)      # for tables
library(gridExtra)      # for tables

# Some customization. You can alter or delete as desired (if you know what you a
re doing).

theme_set(theme_bw())     # change theme for ggplot2/ggformula

knitr::opts_chunk$set(
  tidy = FALSE,     # display code as typed (rather than reformatted)
  size = "small")   # slightly smaller font for code
set.seed(6230)
```

### Task 1
We read in the data
```{r}
mlb18 <- filter(read.csv("https://www.openintro.org/data/csv/mlb_players_18.csv"
), AB >= 100)
gf_density(~HR, data=mlb18)
favstats(~HR, data=mlb18)
```
Results above give that $\mu = 12.38$ and $\sigma = 9.5$.

### Task 2
(a)
```{r}
mean( with(mlb18, sample(HR, size=4)))
```
draws one sample of size 4, and calculates $\overline x$ from it.

(b)
To draw 5000 samples of size 4, and calculate $\overline x$ from each one,
I'll use the `do()` command:
```{r}
manyMeanSS4 <- do(5000) * mean( with(mlb18, sample(HR, size=4)))
head(manyMeanSS4)
```

(c)
The distribution using a density plot
```{r}
gf_density(~mean, data=manyMeanSS4)
```
````

````
(d)
The values we computed (5000 of them) have mean and sd
```{r}
favstats(~mean, data=manyMeanSS4)
```

(e)
We found the mean of this sampling distribution to be 12.39 and the
standard error to be 4.65.  Let's use those values for overlaying
a normal distribution.
```{r}
gf_density(~mean, data=manyMeanSS4) %>% gf_dist("norm", mean=12.34,sd=4.74, colo
r="purple")
```

## Aside
Let me pause here from the tasks to suggest some helpful improvement
which represent a deeper dive into R code.  Some of the motivation is
to provide better labeling.  The rest is to put graphs up together and
on the same scale for the purpose of comparison.  When I produce the resulting g
raphs, in the end, I will use the `grid.arrange()` command,
which requires the **gridExtra** package.

Here are the commands I am using to produce the next plot, which allows
for an easier comparison between the population distribution and the
sampling distribution with $n=4$.
```{r fig.height = 5}
p1 <- gf_dhistogram(~HR, data=mlb18) %>%
  gf_labs(title = "Population distribution") %>%
  gf_refine(scale_x_continuous(limits=c(-5,55)))
p2 <- gf_density(~mean, data=manyMeanSS4) %>%
  gf_labs(title="sampling distribution, with n=4", x="sample means") %>%
  gf_refine(scale_x_continuous(limits=c(-5,55))) %>%
  gf_dist("norm", mean=12.34,sd=4.74, color="purple")
grid.arrange(p1, p2, nrow=2)
```

### Task 4 (skipping Task 3)

```{r fig.height=5}
manyMeanSS180NoReplace <- do(5000) * mean(with(mlb18,sample(HR,size=180)))
manyMeanSS180WithReplace <- do(5000) * mean(with(mlb18,sample(HR,size=180,replac
e=TRUE)))
favstats(~mean, data=manyMeanSS180NoReplace)
favstats(~mean, data=manyMeanSS180WithReplace)
p1 <- gf_density(~mean, data=manyMeanSS180NoReplace) %>%
  gf_labs(title="sampling distribution, n=180, SRS",x="sample means") %>%
  gf_refine(scale_x_continuous(limits=c(9,16))) %>%
  gf_dist("norm", mean=12.39,sd=.54, color="purple")
p2 <- gf_density(~mean, data=manyMeanSS180WithReplace) %>%
  gf_labs(title="sampling distribution, n=180, iid",x="sample means") %>%
  gf_refine(scale_x_continuous(limits=c(9,16))) %>%
  gf_dist("norm", mean=12.37,sd=.706, color="purple")
grid.arrange(p1, p2, nrow=2)
```
````

Sampling Distribution Activity

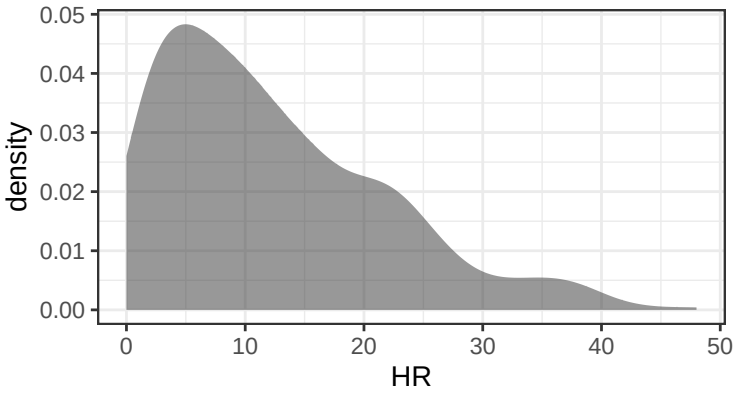
Thomas Scofield

September 21, 2021

Task 1

We read in the data

```
mlb18 <- filter(read.csv("https://www.openintro.org/data/csv/mlb_players_18.csv"), AB >= 100)
gf_density(~HR, data=mlb18)
```




```
favstats(~HR, data=mlb18)
```

```
##  min Q1 median Q3 max     mean       sd   n missing
##    0  5     10 18  48 12.37844 9.500945 436       0
```

Results above give that $\mu = 12.38$ and $\sigma = 9.5$.

Task 2

(a)

```
mean( with(mlb18, sample(HR, size=4)))
```

```
## [1] 11.5
```

draws one sample of size 4, and calculates $\overline{x}$ from it.

(b) To draw 5000 samples of size 4, and calculate $\overline{x}$ from each one, I'll use the `do()` command:

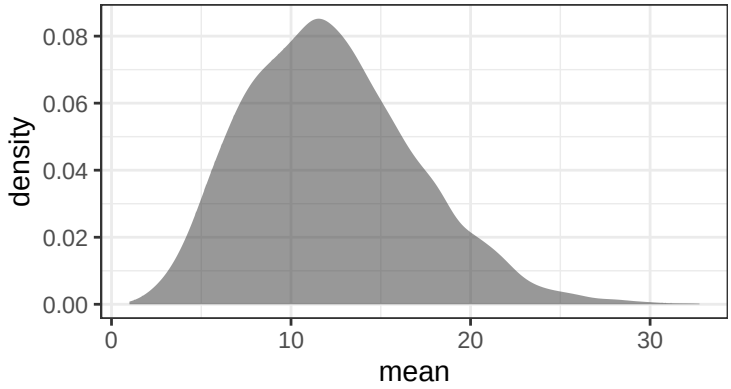
```
manyMeanSS4 <- do(5000) * mean( with(mlb18, sample(HR, size=4)))
head(manyMeanSS4)
```

```
##    mean
## 1  8.75
## 2  4.50
## 3 16.00
```

```
## 4 13.25
## 5  7.50
## 6  8.50
```

(c) The distribution using a density plot

```
gf_density(~mean, data=manyMeanSS4)
```



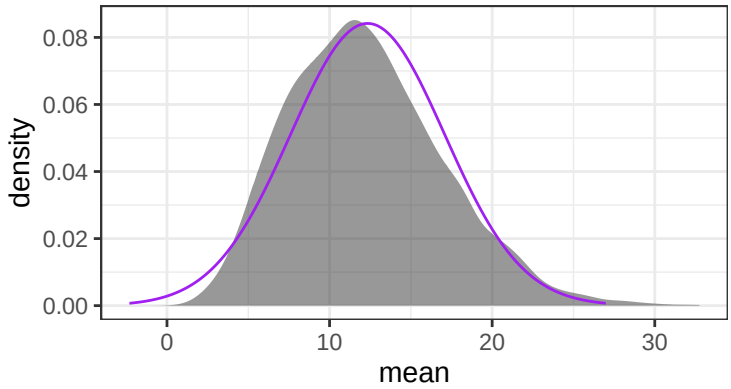
(d) The values we computed (5000 of them) have mean and sd

```
favstats(~mean, data=manyMeanSS4)
```

```
##  min   Q1 median    Q3   max    mean       sd    n missing
##    1 8.75  11.75 15.25 32.75 12.2396 4.743408 5000       0
```

(e) We found the mean of this sampling distribution to be 12.39 and the standard error to be 4.65. Let's use those values for overlaying a normal distribution.

```
gf_density(~mean, data=manyMeanSS4) %>% gf_dist("norm", mean=12.34,sd=4.74, color="purple")
```



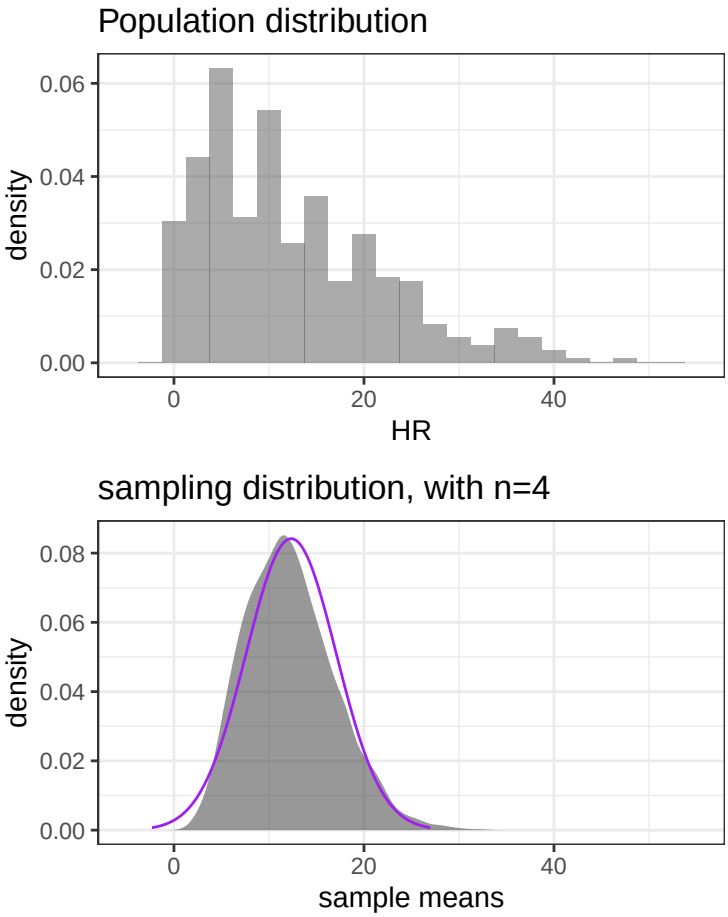
Aside

Let me pause here from the tasks to suggest some helpful improvement which represent a deeper dive into R code. Some of the motivation is to provide better labeling. The rest is to put graphs up together and on the same scale for the purpose of comparison. When I produce the resulting graphs, in the end, I will use the `grid.arrange()` command, which requires the **gridExtra** package.

Here are the commands I am using to produce the next plot, which allows for an easier comparison between the population distribution and the sampling distribution with $n = 4$.

```
p1 <- gf_dhistogram(~HR, data=mlb18) %>%
  gf_labs(title = "Population distribution") %>%
  gf_refine(scale_x_continuous(limits=c(-5,55)))
p2 <- gf_density(~mean, data=manyMeanSS4) %>%
  gf_labs(title="sampling distribution, with n=4", x="sample means") %>%
  gf_refine(scale_x_continuous(limits=c(-5,55))) %>%
  gf_dist("norm", mean=12.34,sd=4.74, color="purple")
grid.arrange(p1, p2, nrow=2)
```

```
## Warning: Removed 2 rows containing missing values (geom_bar).
```



Task 4 (skipping Task 3)

```
manyMeanSS180NoReplace <- do(5000) * mean(with(mlb18,sample(HR,size=180)))
manyMeanSS180WithReplace <- do(5000) * mean(with(mlb18,sample(HR,size=180,replace=TRUE)))
favstats(~mean, data=manyMeanSS180NoReplace)
```

```
##       min       Q1   median       Q3      max     mean        sd    n missing
##  10.40556 12.02222 12.38889 12.74444 14.24444 12.38794 0.5398628 5000       0
```

```
favstats(~mean, data=manyMeanSS180WithReplace)
```

```
##       min       Q1 median       Q3      max     mean        sd    n missing
```

```
##  9.944444 11.88889  12.35 12.85556 14.88333 12.37334 0.7064525 5000       0
```

```
p1 <- gf_density(~mean, data=manyMeanSS180NoReplace) %>%
  gf_labs(title="sampling distribution, n=180, SRS",x="sample means") %>%
  gf_refine(scale_x_continuous(limits=c(9,16))) %>%
  gf_dist("norm", mean=12.39,sd=.54, color="purple")
p2 <- gf_density(~mean, data=manyMeanSS180WithReplace) %>%
  gf_labs(title="sampling distribution, n=180, iid",x="sample means") %>%
  gf_refine(scale_x_continuous(limits=c(9,16))) %>%
  gf_dist("norm", mean=12.37,sd=.706, color="purple")
grid.arrange(p1, p2, nrow=2)
```