Assignment 5

Abe Kazemzadeh[*] OpenIntro Statistics[†]

4/20/2020

Sampling from Ames, Iowa

Sampling from Ames, Iowa: If you have access to data on an entire population, say the size of every house in Ames, Iowa, it's straight forward to answer questions like, "How big is the typical house in Ames?" and "How much variation is there in sizes of houses?". If you have access to only a sample of the population, as is often the case, the task becomes more complicated. What is your best guess for the typical size if you only know the sizes of several dozen houses? This sort of situation requires that you use your sample to make inference on what your population looks like.

Data

Load the ames.RData dataset (same as last homework).

```
load("ames.RData")
```

We'll start with a simple random sample of size 60 from the population. Note that the data set has information on many housing variables, but initially we'll focus on the size of the house, represented by the variable Gr.Liv.Area.

First, set the random seed. This will initialize the state of the random number generator so that it is the same for the whole class (631 is an arbitrary value):

```
set.seed(631)
population <- ames$Gr.Liv.Area
samp <- sample(population, 60)
```

Confidence intervals

Confidence intervals: One of the most common ways to describe the typical or central value of a distribution is to use the mean. In this case we can calculate the mean of the sample using

```
sample_mean <- mean(samp)
sample_mean
```

[*]University of St. Thomas, abe.kazemzadeh@stthomas.edu
[†]Creative Commons–Share Alike By Attribution, https://www.openintro.org/license

```
## [1] 1408.7
```

Based only on this single sample, the best estimate of the average living area of houses sold in Ames would be the sample mean, usually denoted as $\bar{x}$ (in LaTeX embedded in RMarkdown this is written as `$\bar x$` and as an R variable we're calling it `sample_mean`). That serves as a good point estimate but it would be useful to also communicate how uncertain we are of that estimate. This can be captured by using a confidence interval.

We can calculate a 95% confidence interval for a sample mean by adding and subtracting 1.96 standard errors to the point estimate. Remember, ± 1.96 comes from `qnorm(.025)` and `qnorm(.975)`.

```r
se <- sd(samp)/sqrt(60)
lower <- sample_mean - 1.96 * se
upper <- sample_mean + 1.96 * se
c(lower, upper)
```

```
## [1] 1307.728 1509.672
```

This is an important inference that we've just made: even though we don't know what the full population looks like, we're 95% confident that the true average size of houses in Ames lies between the values lower and upper. There are a few conditions that must be met for this interval to be valid.

Question 1

For the confidence interval to be valid, the sample mean must be normally distributed and have standard error s/sqrt(n). Which of the following is not a condition needed for this to be true?

- (a) The sample is random.
- (b) The sample size, 60, is less than 10% of all houses.
- (c) The sample distribution must be nearly normal.

Answer : (c)

Question 2

What does "95% confidence" mean?

- (a) 95% of the time the true average area of houses in Ames, Iowa, will be in this interval.
- (b) 95% of random samples of size 60 will yield confidence intervals that contain the true average area of houses in Ames, Iowa.
- (c) 95% of the houses in Ames have an area in this interval.
- (d) 95% confident that the sample mean is in this interval.

Answer: a

Question 3

Does your confidence interval capture the true average size of houses in Ames? I.e.,

```
mean(population)
```

```
## [1] 1499.69
```

Testing the Theory

Using R, we're going to recreate many samples to learn more about how sample means and confidence intervals vary from one sample to another. Loops come in handy here. Here is the rough outline:

1) Obtain a random sample.
2) Calculate the sample's mean and standard deviation.
3) Use these statistics to calculate a confidence interval.
4) Repeat steps (1)-(3) 50 times.

But before we do all of this, we need to first create empty vectors where we can save the means and standard deviations that will be calculated from each sample. And while we're at it, let's also store the desired sample size as n.

```
samp_mean <- rep(NA, 50)
samp_sd <- rep(NA, 50)
n <- 60
```

Now we're ready for the loop where we calculate the means and standard deviations of 50 random samples.

```
set.seed(123) # initialize random number generator
for(i in 1:50){
   samp <- sample(population, n) # obtain a sample of size n = 60 from the population
   samp_mean[i] <- mean(samp)    # save sample mean in ith element of samp_mean
   samp_sd[i] <- sd(samp)        # save sample sd in ith element of samp_sd
}
```

Lastly, we construct the confidence intervals.

```
lower <- samp_mean - 1.96 * samp_sd/sqrt(n)
upper <- samp_mean + 1.96 * samp_sd/sqrt(n)
```

Lower bounds of these 50 confidence intervals are stored in lower, and the upper bounds are in upper. Let's view the first interval.

```
c(lower[1], upper[1])
```

```
## [1] 1390.046 1677.388
```

Now, we'll use a function `plot_ci` which is stored in the 'ames.RDatafile, but we could also copy-and-paste the function from https://rdrr.io/github/andrewpbray/oilabs/src/R/plot__ci.R , e.g.,

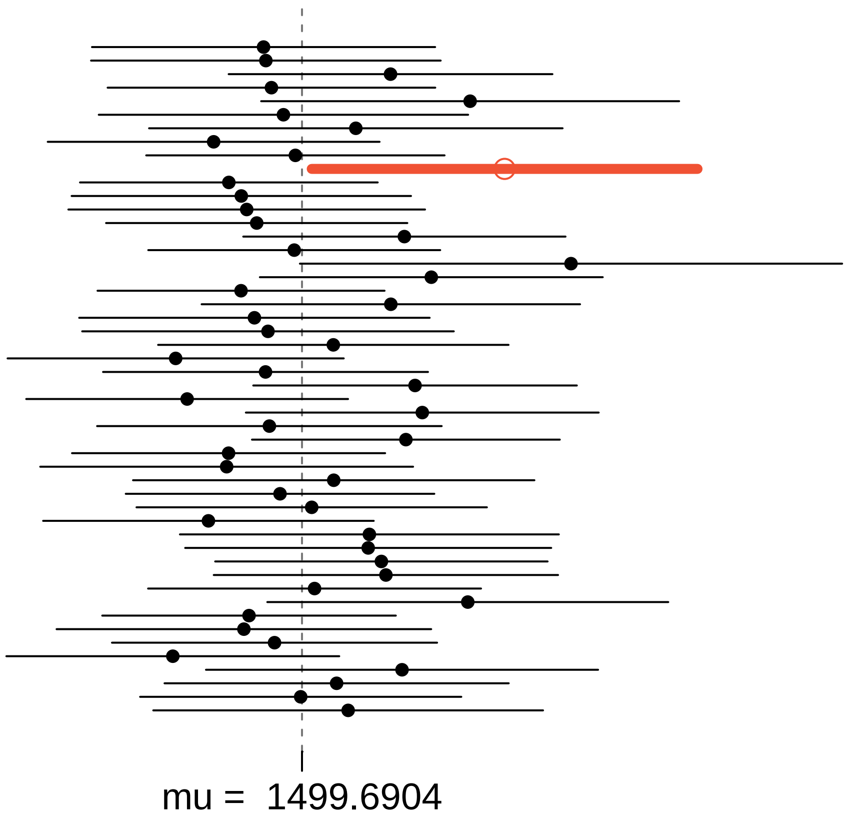
```r
plot_ci <- function(lo, hi, m) {
  par(mar=c(2, 1, 1, 1), mgp=c(2.7, 0.7, 0))
  k <- length(lo)
  ci.max <- max(rowSums(matrix(c(-1*lo,hi),ncol=2)))

  xR <- m + ci.max*c(-1, 1)
  yR <- c(0, 41*k/40)

  plot(xR, yR, type='n', xlab='', ylab='', axes=FALSE)
  abline(v=m, lty=2, col='#00000088')
  axis(1, at=m, paste("mu = ",round(m,4)), cex.axis=1.15)
  #axis(2)
  for(i in 1:k){
    x <- mean(c(hi[i],lo[i]))
    ci <- c(lo[i],hi[i])
    if((m < hi[i] & m > lo[i])==FALSE){
      col <- "#F05133"
      points(x, i, cex=1.4, col=col)
      #       points(x, i, pch=20, cex=1.2, col=col)
      lines(ci, rep(i, 2), col=col, lwd=5)
    } else{
      col <- 1
      points(x, i, pch=20, cex=1.2, col=col)
      lines(ci, rep(i, 2), col=col)
    }
  }
}
```

Then run `plot_ci` with our upper and lower bounds together with the population mean

```r
plot_ci(lower, upper, mean(population))
```



Question 4

How many confidence intervals do not contain the true population mean?
Answer: 1(as one in red line)

Question 5

Question 5 [MULTIPLE CHOICE] What is the appropriate critical value for a 99% confidence level?

- (a) 0.01
- (b) 0.99
- (c) 1.96
- (d) 2.33
- (e) 2.58

```
p99<- 1-(1-0.99)/2
qnorm(p99)
```

```
## [1] 2.575829
```

Answer for Question 5 is : 2.58, (e)

Question 6

Calculate 50 confidence intervals at the 99% confidence level. Do not obtain new samples. Simply calculate new intervals based on the sample means and standard deviations you have already collected. Using the plot_ci function, plot all intervals and calculate the proportion of intervals that include the true population mean. What proportion do you get? Use a seed of 123.

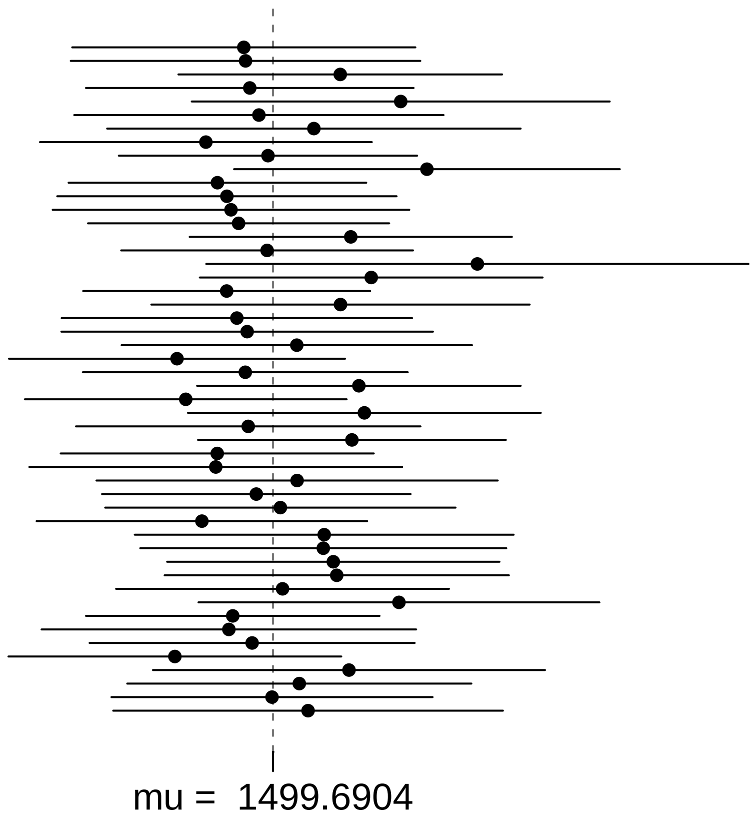
```r
lower99 <- samp_mean - 2.58 * samp_sd/sqrt(n)
upper99 <- samp_mean + 2.58 * samp_sd/sqrt(n)


c(lower99[1], upper99[1])
```

```
## [1] 1344.599 1722.835
```

Using plot_ci function, plotting all intervals:

```r
plot_ci(lower99, upper99, mean(population))
```



# Answer to Question6 is 100%

Extra Question 1

Go through each line of `plot_ci` and briefly describe what it does. Use the line numbers from https://pastebin.com/GUDjgjWe to indicate which line you are describing.

Answer:
line 2: plot_ci <- function(lo, hi, m) {
-> creating function named plot_ci with 3 parameters called low,hi,m

line 3: par(mar=c(2, 1, 1, 1), mgp=c(2.7, 0.7, 0))

-> here par sets or adjusts plotting parameters with two parameters mar and mgp. mar – A numeric vector of length 4, which sets the margin sizes in the following order: bottom, left, top, and right. mgp – A numeric vector of length 3, which sets the axis label locations relative to the edge of the inner plot window. The first value represents the location the labels (i.e. xlab and ylab in plot), the second the tick-mark labels, and third the tick marks. so, here we are specifying the margin and axis label locations in par.

line 4: k <- length(lo)
-> assigning length of low to k variable.

line5: ci.max <- max(rowSums(matrix(c(-1*lo,hi),ncol=2)))*
*-> assigns the sum of both rows in the matrix created from -1*low and hi values into ci.max variables.

line7: xR <- m + ci.max*c(-1, 1)
-> assigns value calculated from right hand side to xR variable.

line8: yR <- c(0, 41*k/40)
-> assign vector of values to variable yR

line10: plot(xR, yR, type='n', xlab='', ylab='', axes=FALSE)
-> plot for xR and yR with no title on xaxis and yaxis and type is set to no plotting

line11: abline(v=m, lty=2, col='#00000088')
-> this adds straight line to the plot where v is x value for vertical line and its set to m value that is passed in the function with line type as dashed(lty =2) and color with '#00000088'

line12: axis(1, at=m, paste("mu =",round(m,4)), cex.axis=1.15)
-> adding axis to a plot where 1 means below at at = m(tick-marks are to be drawn where at = m) with label as value displaying value of m after rounding to 4 decimal places with showing as mu = m value.cex.axis – Specify the size of the tick label numbers/text with a numeric value of length 1.15.

line14: for(i in 1:k){
-> for loop is initiated, and looks i is between 1 and k

line15: x <- mean(c(hi[i],lo[i]))
-> it assigns the mean of vector values of each i to x variable

line16:ci <- c(lo[i],hi[i])
-> it assigns the vector values of each i to ci variable

line17: if((m < hi[i] & m > lo[i])==FALSE){
-> it is checking for condition, if m is less than the each value that is i in hi and m is greater than each value in lo is false then execute the below statement

line18: col <- "#F05133"
-> assigning color to col

line19: points(x, i, cex=1.4, col=col)
-> this generates the sequence of probability points for values of x,i with tick label length of 1.4 and and color from col variable

line21:lines(ci, rep(i, 2), col=col, lwd=5)
-> this creates Connected Line Segments to a Plot for ci value by replicating values of i for 2 times with color in col variable and line width of 5

line22: } else{
-> generates else if above condition is True

line23: col <- 1
-> assigns 1 to col variable

line24: points(x, i, pch=20, cex=1.2, col=col)
->this generates the sequence of probability points for values of x,i with tick label length of 1.2 and color from col variable and it creates a point as we specifiedd pch = 20

line25: lines(ci, rep(i, 2), col=col)
-> this creates Connected Line Segments to a Plot for ci value by replicating values of i for 2 times with color in col variable and line width of 5.

line26: }
-> closing parenthesis for else

line27: }
-> closing parenthesis for for loop

line28: }
-> closing parenthesis for function.

T-Distribution

Depending on your data collection, it may be hard to get a sample of size 60. One issue we will have if we have smaller sample is that we will need to use the Student's t-distribution. The sample standard deviation in particular is the calculation that will require the t-distribution because the t-distribution has "fatter tails" when the degrees-of-freedom parameter is low. Let's say we have a sample of size 15. If we want to find the critical values for 95% confidence level using a 14 degrees-of-freedom t-distribution ($df = n - 1$), we would use $t^* = 2.14$ instead of $z^* = 1.96$

```
qt(.025, 14)
```

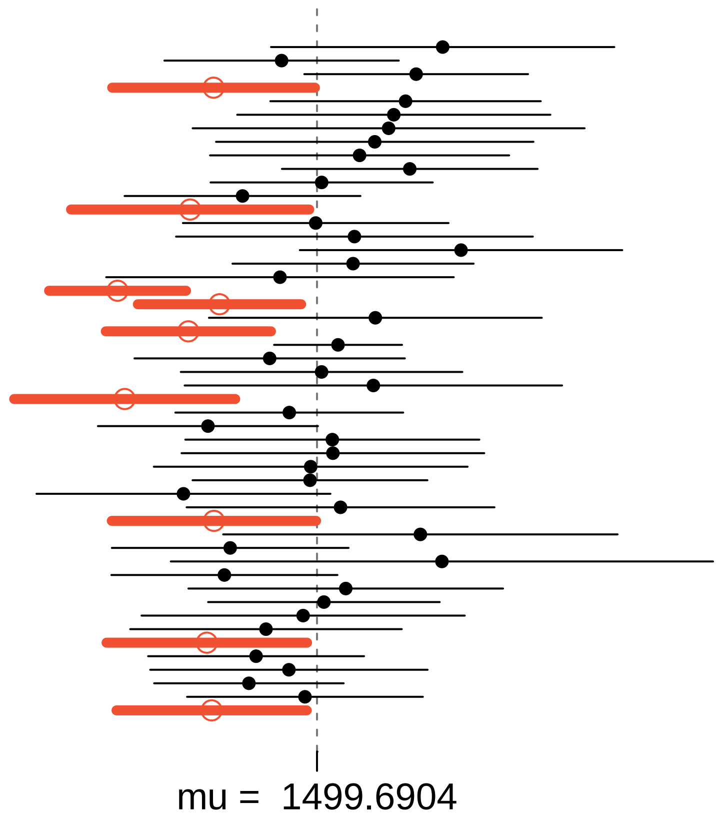
```
## [1] -2.144787
```

```
qt(.975, 14)
```

```
## [1] 2.144787
```

To show this, let's rerun our earlier simulation with $n = 15$ instead of 60:

```
samp_mean <- rep(NA, 50)
samp_sd <- rep(NA, 50)
n <- 15
set.seed(123)
for(i in 1:50){
   samp <- sample(population, n)  # obtain a sample of size n = 60 from the population
   samp_mean[i] <- mean(samp)     # save sample mean in ith element of samp_mean
   samp_sd[i] <- sd(samp)         # save sample sd in ith element of samp_sd
}
lower <- samp_mean - 1.96 * samp_sd/sqrt(n)
upper <- samp_mean + 1.96 * samp_sd/sqrt(n)
plot_ci(lower, upper, mean(population))
```



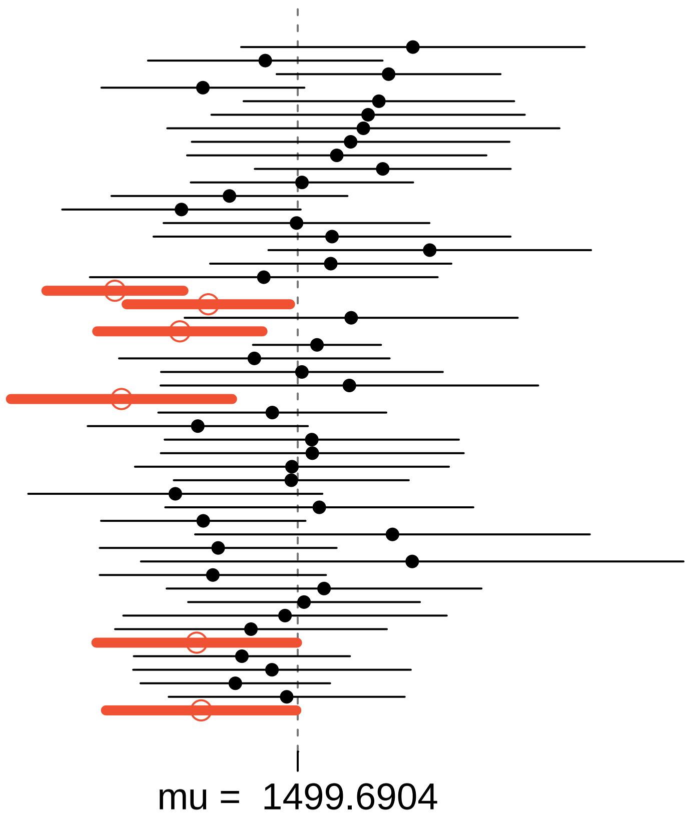
Question 7

How many confidence intervals do not contain the true population mean now?
Answer for Question 7 is : 9

Question 8

We can see that the small sample size statistically led to more intervals that did not contain the population mean. Use the t-distribution to find critical values and then re-run the simulation and see if it fixes the problem, i.e. fewer intervals that do not contain the population mean.

```
samp_mean <- rep(NA, 50)
samp_sd <- rep(NA, 50)
n <- 15
set.seed(123)
for(i in 1:50){
   samp <- sample(population, n)  # obtain a sample of size n = 60 from the population
   samp_mean[i] <- mean(samp)     # save sample mean in ith element of samp_mean
   samp_sd[i] <- sd(samp)         # save sample sd in ith element of samp_sd
}
lowert95 <- samp_mean - 2.14 * samp_sd/sqrt(n)
uppert95 <- samp_mean + 2.14 * samp_sd/sqrt(n)
plot_ci(lowert95, uppert95, mean(population))
```



Submission

I'm trying a different format for the homework this time. The goals are to make you more familiar with RMarkdown, get a taste of GitHub, to learn from peers, and to get specific feedback more rapidly. To try to achieve this, the new format is to create one or more discussion posts related to the homework. The constraints of the post are as follows. You should post at least once and your post should be categorized into one of the following categories:

- Question: something you're uncertain about or don't understand. Post the question and try to phrase it in a way that will help others answer you.
- Response: respond to a question. Try to phrase the answer so that it's helpful to the asker's learning, so give more than just the answer but a bit of explanation, background info, an example, etc.
- Recommendation: did you have an insight from the homework, find something useful in the book or on the web, or try some variation on the something in the homework? Sharing this would be an example of a recommendation that you could post.
- Summary: imagine someone outside the class who doesn't know much or anything about statistics (like your mom, your child, a neice or nephew, etc.). Explain to them the main idea of the homework or a specific part of it.

Make sure that your question, response, recommendation, or learning summary is on the topics in the homework, especially if you just do one post (if you do more than one post, feel free to take more leeway on the later posts). On-topic subjects can include RMarkdown and GitHub as well as the theoretical topics.

Use the subject “hw5 , ” where is “question”, “response”, “recommendation”, or “summary” and is a brief description.

Submit the link(s) to the discussion post that you created or commented on.

I hope this variation on the assignment is fun or at least an interesting change of pace for you. For me it is these things and also an attempt to adapt the class to the new online format.