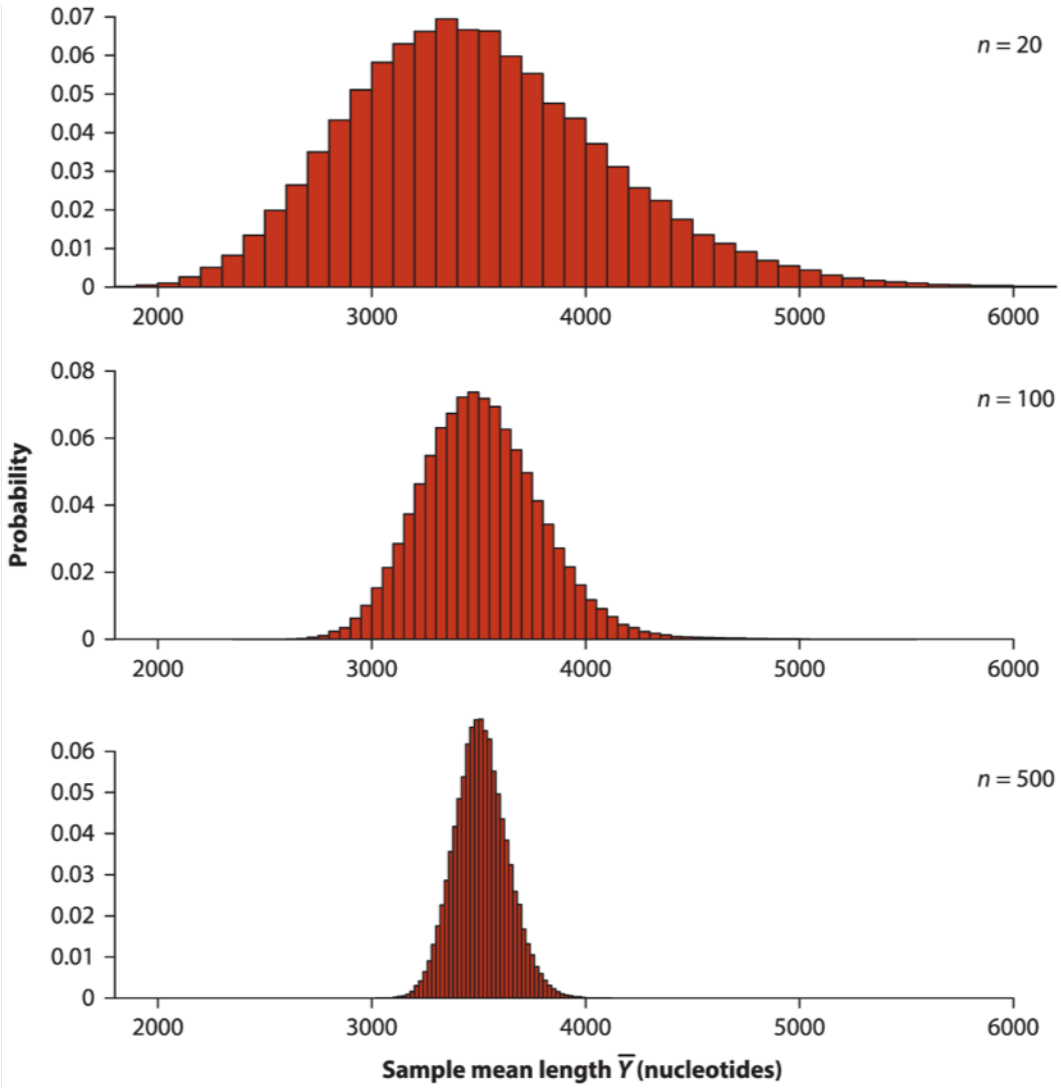
R Lab #5c - Simulating sampling distributions

Contents

The sampling distribution of gene lengths with different sample sizes

The human genome has 22,385 genes of known lengths, with a population mean gene length of 3511.457 bp. This is a parameter because we have full knowledge of all gene lengths. The following figure (4-1.4) shows the sampling distribution for the mean with samples of different sizes. Since sampling a population introduces sample error the estimate mean is different than the parameter mean, but increasing the sample size increases the precision of the estimate.



Whitlock & Schluter, *The Analysis of Biological Data*, 3e © 2020 W. H. Freeman and Company

The authors obtained these sampling distributions by simulating many samples for each size, calculating the mean of each sample, and graphing the distribution of means. Let's try to mimic this in R by creating 1000 samples...

Using a loop to simulate many samples with n = 20

To do this we need to think like a computer. We need to repeat 1000 times a) randomly sample n gene lengths, b) calculate mean, and c) append mean to vector.

First we import the data. The file is on Canvas and can be downloaded to your working directory. See Modules > Data sets for examples/questions in textbook > Chapter 4 data sets > chap04e1HumanGeneLengthsLongestTranscript.csv

```
geneData <- read.csv("chap04e1HumanGeneLengthsLongestTranscript.csv")
```

Next we can establish a place to store the means when the sample size is 20. You can create an empty vector with c():

```
n20 <- c()
```

Now we need to set up the loop.

The first part tells R how many times you want to do the loop. The syntax is *for (i in 1:1000)*. The first time through the loop i=1, then i=2, and so on until i=1000.

The second part of the loop is in {} and tells R what you want to do in each iteration of the loop. We can do our task by using the commands mean() and sample(). The command calculates the mean from a sample and adds it to the vector n20.

What does sample() do? This command will subsample a vector. For arguments it needs the vector (x) and the number of samples that you want to draw from that vector (size). You can also sample with (sampled item is returned before next sample drawn) or without (items cannot be sampled twice) replacement. Since we do not want to sample the same gene twice we will run it with replace=F.

```
for (i in 1:1000) {
    n20[i] <- mean(sample(geneData$size, 20, replace = F))
}
```

Remember that each time through the loop i increases by one. So on the first loop n20[1] is adding the first number to the vector, second time through n20[2] adds the second value, etc. until you get to n20[1000].

Now the vector n20 has a length of 1000, with each value the mean of 20 randomly sampled gene lengths. We can graph the sampling distribution with hist():

Repeating the loop with other sample sizes

n = 100

```
n100 <- c()
for (i in 1:1000) {
    n100[i] <- mean(sample(geneData$size, 100, replace = F))
}
```

n = 500

```
n500 <- c()
for (i in 1:1000) {
    n500[i] <- mean(sample(geneData$size, 500, replace = F))
}
```

Now we have three vectors with 1000 means when the sample size is 20 (n20), 100 (n100), and 500 (n500).

Making a graph with multiple panels

Now let's try to make a figure with all three histograms like in the book.

First ggplot needs data to be in a "data frame", and currently we just have three vectors. There are different ways to do this, but I'd like to make a data from with 2 columns (sampleSize and means) and 3000 rows (1000 samples x 3 different sampling size). In this case all of the means are in one column (means), but can be split into different groups using another column (sampleSize).

Again, multiple ways to do this but this is how I would do it...

First I will create the sampleSize column. Here I use c() command to concatenate and the rep() command to repeat something. In this case rep is used to repeat a string 1000 times:

```
sampleSize <- c(rep("n20", 1000), rep("n100", 1000), rep("n500",
    1000))
```

Now make the column means using c() to concatenate the 1000 means we generated for each sample size:

```
means <- c(n20, n100, n500)
```

Now we'll put these together in a data frame called "samples":

```
samples <- data.frame(sampleSize, means)
```

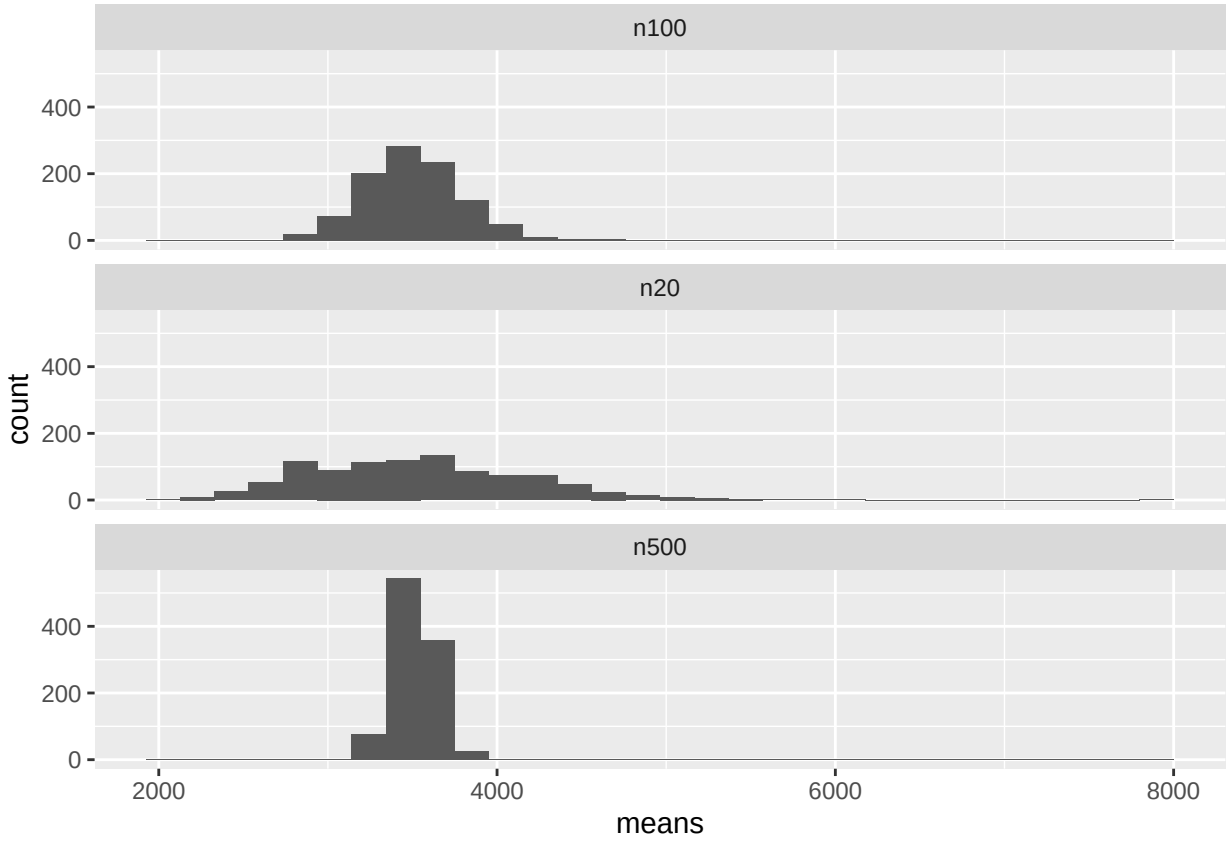
Now we'll use ggplot to make something that looks like Fig 4.1-4, with stacked histograms of sampling distributions of the mean with different sample sizes.

Recall that making a graph with ggplot takes at least two commands: one to tell it which data to graph and another to tell it what type of graph to make.

Here we want to make three histograms (geom_histogram) stacked, with the info in column sampleSize used to split data into the three plots. The last part is done by adding the command facet_wrap(). This tells R which column to use to split data into different plots, and also to do multiple rows, columns, or both.

This is the command to make the three plots stacked into one column:

```
library(ggplot2)
ggplot(samples, aes(x = means)) + geom_histogram() + facet_wrap(~sampleSize,
    ncol = 1)
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



So that should produce three stacked histograms. Note that your histograms might look different than what is shown here because we generated different random samples.

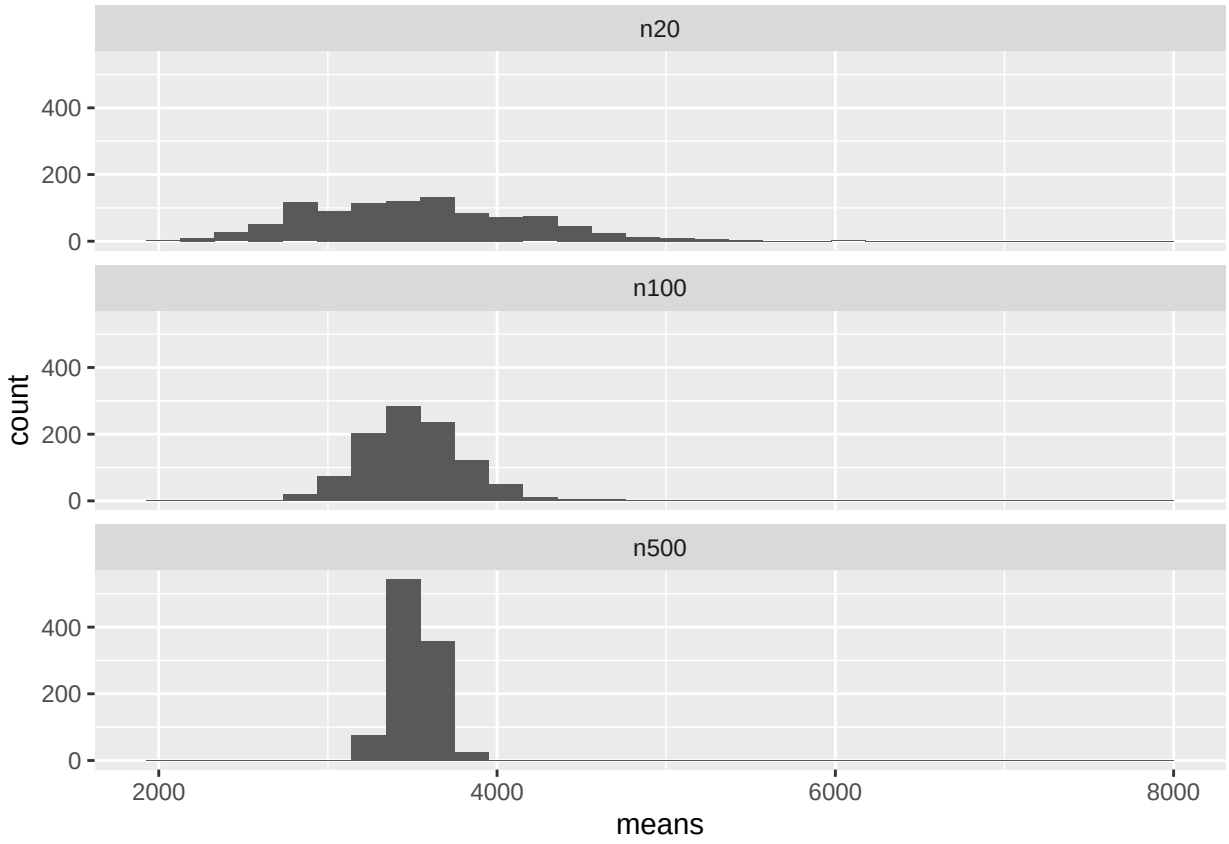
We are on the right track, but the graph needs improvement...

First, notice that n100 is on top rather than n20. This is because ggplot is doing it by alpha-numeric order (and 1 comes before 2). We can fix this by changing the column sampleSize into a "factor" with different "levels." In the R language, "factor" is the term for a categorical variable, and "level" is the term for a category/group within the variable. We can use the factor() function to change the format of this variable and use the levels option to override the default order of categories (to n50, n100, and then n500).

```
samples$sampleSize <- factor(samples$sampleSize, levels = c("n20",
    "n100", "n500"))
```

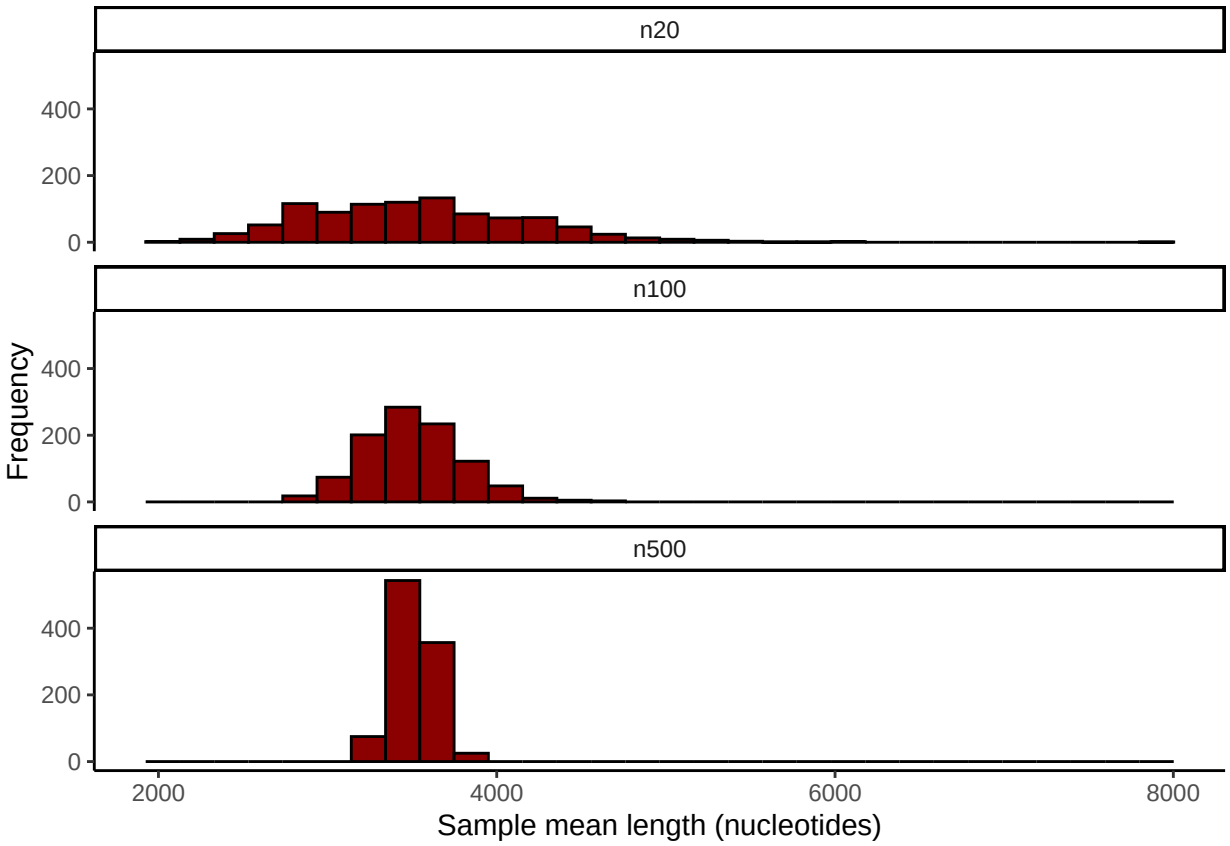
Now try to plot again:

```
library(ggplot2)
ggplot(samples, aes(x = means)) + geom_histogram() + facet_wrap(~sampleSize,
    ncol = 1)
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Now let's change the axis labels (xlab and ylab commands), make the bars have black border and red fill (add color and fill arguments to geom_histogram), and simplify the look (them_classic command).

```
library(ggplot2)
ggplot(samples, aes(x = means)) + geom_histogram(color = "black",
    fill = "darkred") + facet_wrap(~sampleSize, ncol = 1) + xlab("Sample mean length (nucleotides)") +
    ylab("Frequency") + theme_classic()
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Looking better! More tweaks can be made but that is good enough for now. Each plot is showing a sampling distribution of the mean with 1000 samples, but with a different number of genes selected in each sample. As this number increases the sampling distribution becomes more narrow, and closes in on the true parameter mean.