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

We consider real estate data from the city of Ames, Iowa. The details of every real estate transaction in Ames is recorded by the City Assessor's office. Our particular focus for this lab will be all residential home sales in Ames between 2006 and 2010. This collection represents our population of interest. In this lab we would like to learn about these home sales by taking smaller samples from the full population. Let's load the data.

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

We see that there are quite a few variables in the data set, enough to do a very in-depth analysis. For this lab, we'll restrict our attention to just two of the variables: the above ground living area of the house in square feet (`Gr.Liv.Area`) and the sale price (`SalePrice`). To save some effort throughout the lab, create two variables with short names that represent these two variables.

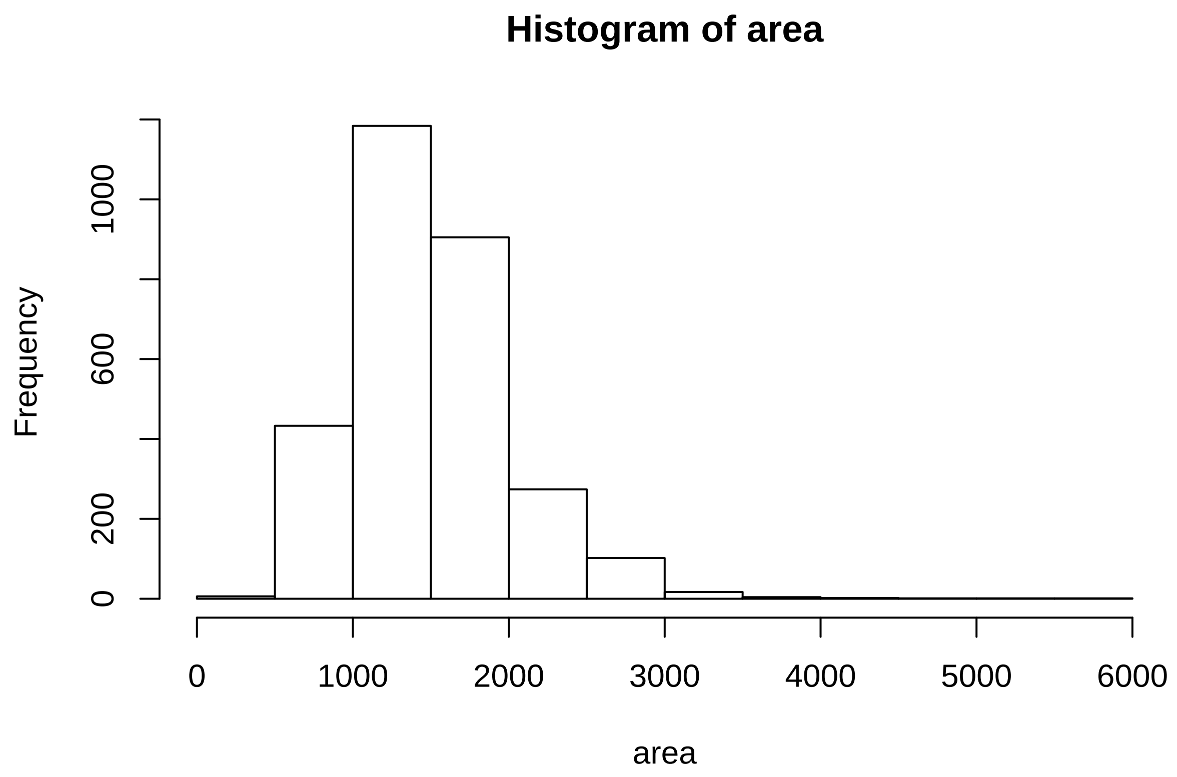
```
area <- ames$Gr.Liv.Area
price <- ames$SalePrice
```

Let's look at the distribution of area in our population of home sales by calculating a few summary statistics and making a histogram.

```
summary(area)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     334    1126    1442    1500    1743    5642
```

```
hist(area)
```



1. Describe this population distribution.

   The population distribution is right skewed and unimodal. The mean area of homes sold in this dataset is at 1500 square feet, and the median area of homes sold in this dataset is 1442 square feet.

The unknown sampling distribution

In this lab we have access to the entire population, but this is rarely the case in real life. 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 living area in Ames based on a sample, we can use the following command to survey the population.

```
samp1 <- sample(area, 50)
```

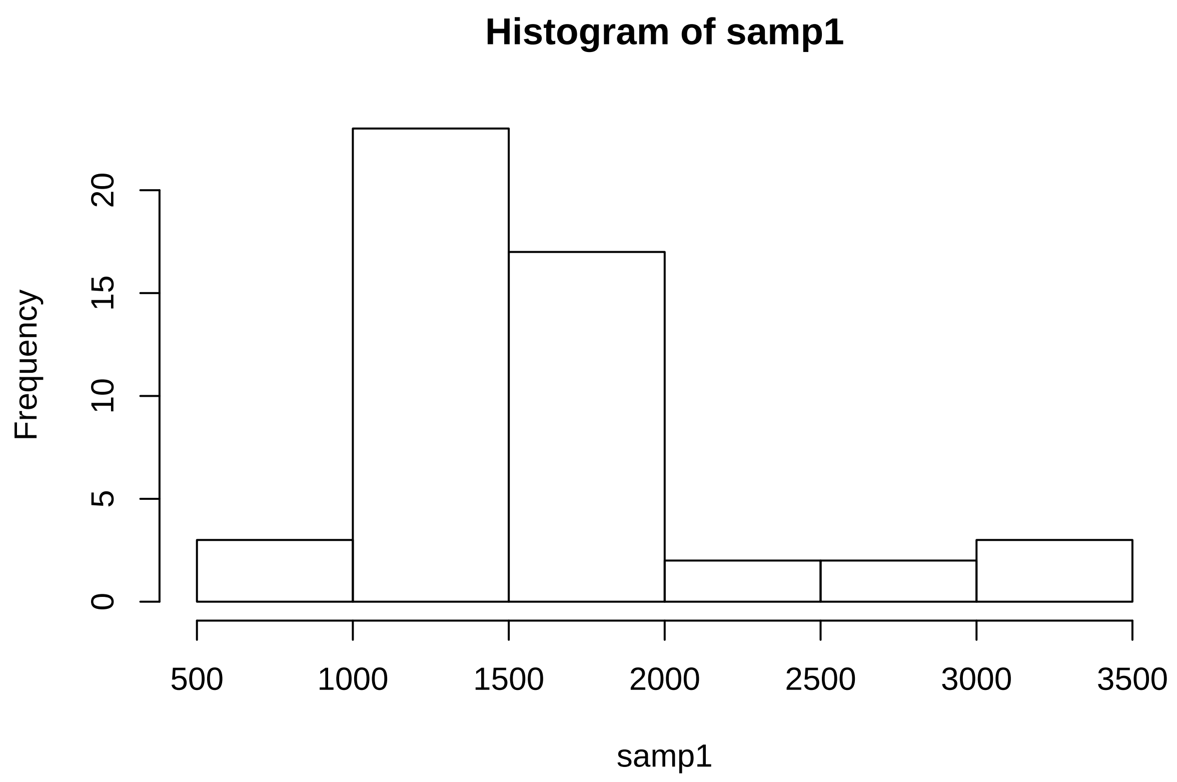
This command collects a simple random sample of size 50 from the vector `area`, which is assigned to `samp1`. This is like going into the City Assessor's database and pulling up the files on 50 random home sales. Working with these 50 files would be considerably simpler than working with all 2930 home sales.

2. Describe the distribution of this sample. How does it compare to the distribution of the population?

```r
# using R to create summary statistics and plotting the distribution
# of the sample
summary(samp1)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##     816    1329    1479    1640    1802    3395
```

```r
hist(samp1)
```



The distribution of this sample of 50 houses is also right skewed and unimodal. The median of the sample is 1479 square feet and the sample mean is 1640 square feet. These are different from the population sample mean and median.

If we're interested in estimating the average living area in homes in Ames using the sample, our best single guess is the sample mean.

```r
mean(samp1)
```

```
## [1] 1639.66
```

Depending on which 50 homes you selected, your estimate could be a bit above or a bit below the true population mean of 1499.69 square feet. In general, though, the sample mean turns out to be a pretty good estimate of the average living area, and we were able to get it by sampling less than 3% of the population.

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?

   First, I took a sample from the area vector:

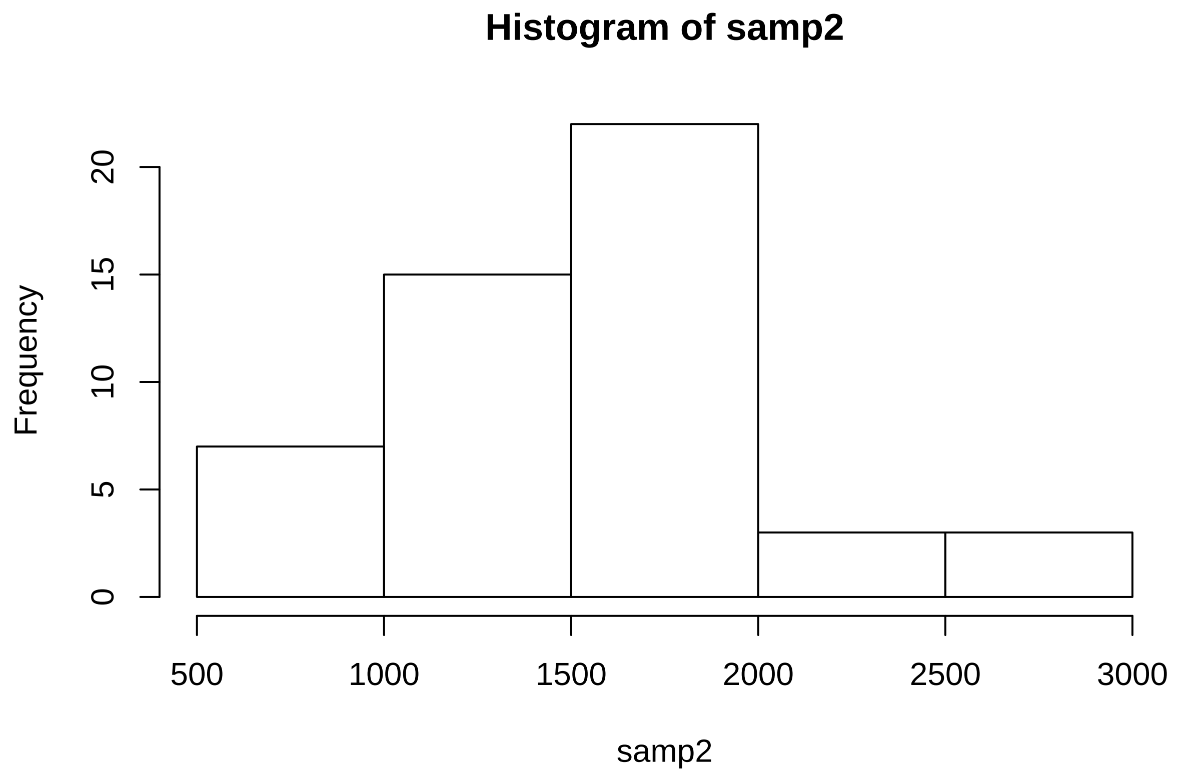
```
samp2 <- sample(area, 50)

# to test the last part of the question
samp3 <- sample(area, 100)
samp4 <- sample(area, 1000)
```

Next, I plotted the distribution and found the summary statistics for this sample.

```
summary(samp2)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##     520    1242    1578    1526    1724    2872
```

```
hist(samp2)
```



```
mean(samp2)
```

```
## [1] 1525.54
```

The sample mean for samp2 differs from samp1. To answer the final part of the question, I believe that having a larger sample size will provide a more accurate estimation of the population mean. This is confirmed when looking at the R output below - the sample of 1000 has a mean that is closer to the population mean.

```
paste('Sample of 100 mean: ', mean(samp3))
```

```
## [1] "Sample of 100 mean:  1551.24"
```

```
paste('Sample of 1000 mean: ', mean(samp4))
```

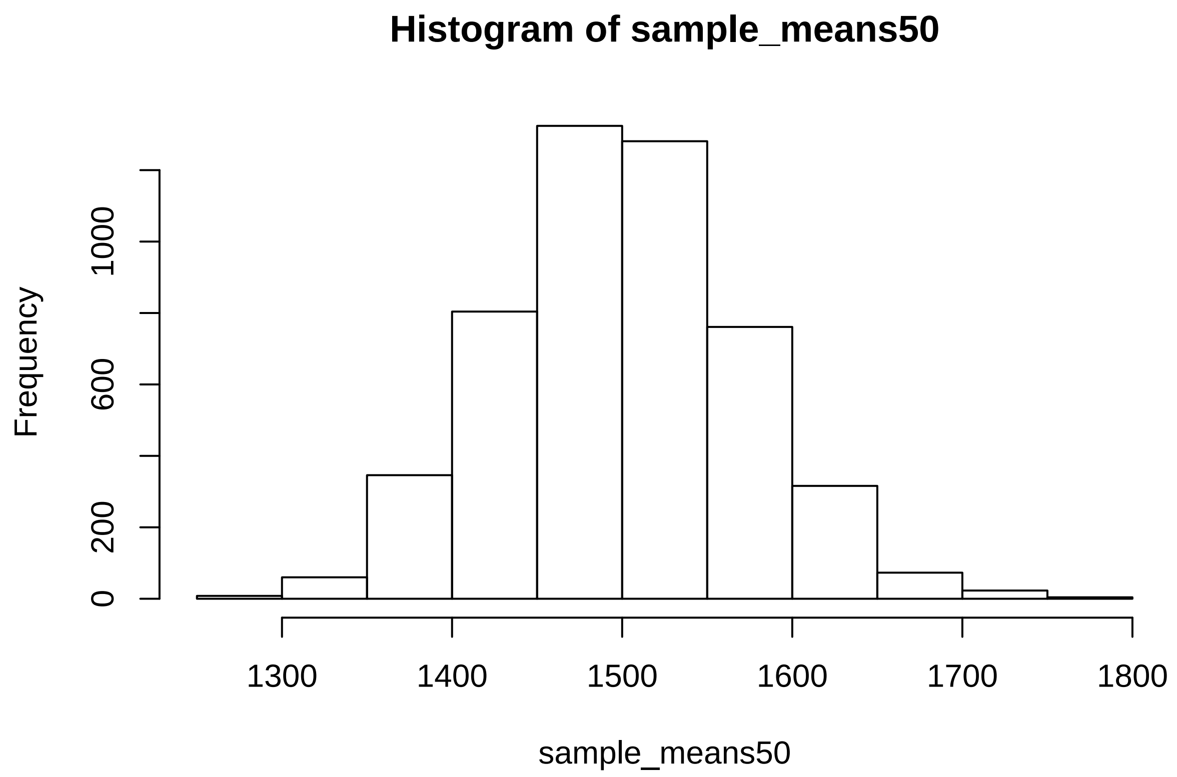
```
## [1] "Sample of 1000 mean:  1505.749"
```

```
paste('Population mean: ', mean(area))
```

```
## [1] "Population mean:  1499.69044368601"
```

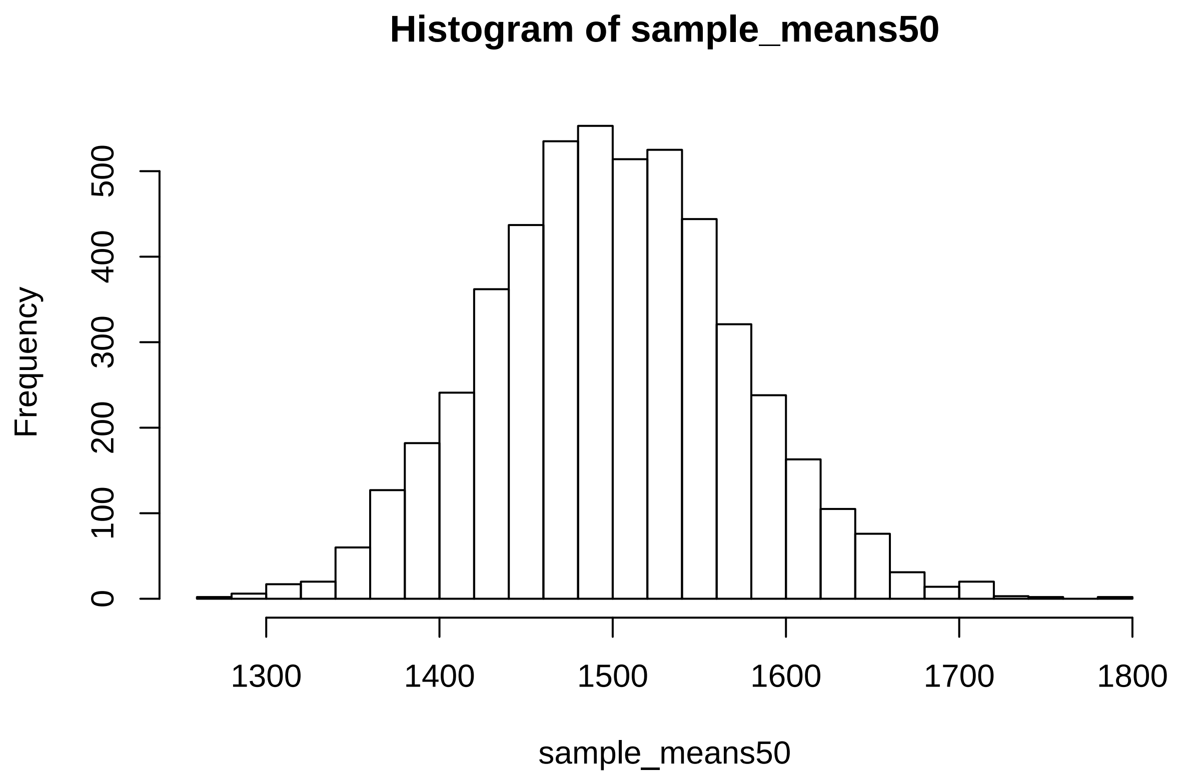
Not surprisingly, every time we take another random sample, we get a different sample mean. It's useful to get a sense of just how much variability we should expect when estimating the population mean this way. The distribution of sample means, called the *sampling distribution*, can help us understand 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(NA, 5000)

for(i in 1:5000){
   samp <- sample(area, 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 = 25)
```



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`. On the next page, we'll review how this set of code works.

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 collected 50,000 sample means?

There are 5,000 elements in sample_means50. The sampling distribution is very close to normal, with a mean that appears to be right around the population mean of 1499.69 square feet. I would expect the sample distribution to become even more normalized if we were to instead collect 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 your first `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 `area` then save the mean of that sample into the `sample_means50` vector. Without the `for` loop, this would be painful:

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

samp <- sample(area, 50)
```

```r
sample_means50[1] <- mean(samp)

samp <- sample(area, 50)
sample_means50[2] <- mean(samp)

samp <- sample(area, 50)
sample_means50[3] <- mean(samp)

samp <- sample(area, 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.

```r
sample_means50 <- rep(NA, 5000)

for(i in 1:5000){
   samp <- sample(area, 50)
   sample_means50[i] <- mean(samp)
   # print(i) --commenting this out to save space
   }
```

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 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 `area`, take its mean, and store it as the *i*th 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.

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 `area` 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?

   ```r
   sample_means_small <- rep(NA, 100)

   for(i in 1:100){
      samp <- sample(area, 50)
      sample_means_small[i] <- mean(samp)
   ```

```
}

sample_means_small
```

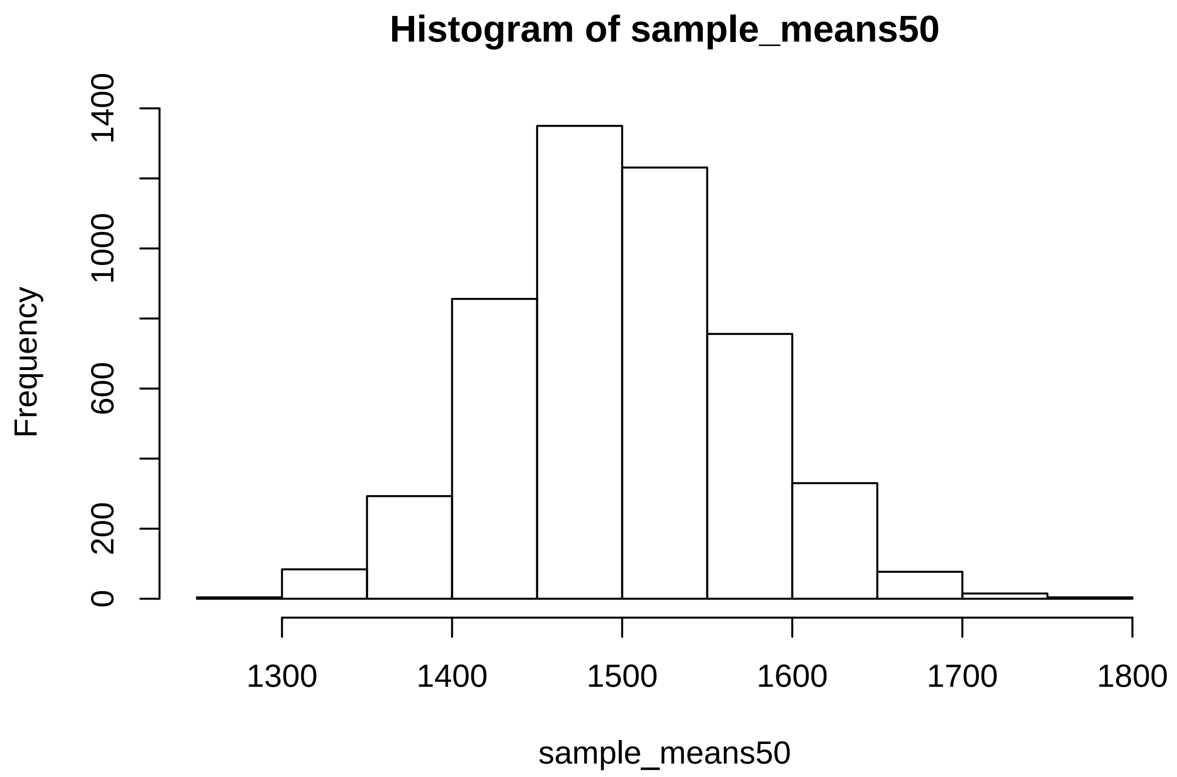
```
##   [1] 1513.42 1512.50 1483.14 1525.38 1389.54 1428.98 1502.80 1442.52
##   [9] 1511.48 1583.34 1386.24 1521.50 1408.46 1501.82 1430.10 1494.68
##  [17] 1497.84 1369.90 1554.60 1522.80 1622.66 1469.04 1488.22 1469.84
##  [25] 1464.58 1473.78 1530.04 1441.12 1482.76 1532.50 1562.34 1567.92
##  [33] 1495.72 1530.30 1505.90 1442.46 1432.74 1572.20 1582.54 1568.02
##  [41] 1432.68 1500.78 1486.38 1491.86 1533.68 1546.44 1572.12 1475.52
##  [49] 1470.44 1456.52 1531.82 1552.12 1511.54 1523.96 1597.78 1480.86
##  [57] 1567.72 1461.98 1439.04 1361.66 1513.20 1476.18 1443.90 1462.28
##  [65] 1560.22 1514.18 1400.20 1441.22 1650.86 1507.04 1392.08 1659.92
##  [73] 1365.74 1397.60 1473.78 1435.28 1564.30 1323.86 1505.20 1568.90
##  [81] 1737.56 1599.94 1536.82 1395.56 1616.68 1511.76 1536.54 1519.02
##  [89] 1604.62 1435.54 1544.70 1607.00 1462.18 1510.52 1491.76 1412.76
##  [97] 1564.80 1570.38 1560.68 1459.62
```

After creating this for loop, I found that there are 100 elements in the sample__means__small object. Each element represents a sample mean of the area dataset based on 50 samples taken to calculate each of the 100 sample means.

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)
```



The sampling distribution that we computed tells us much about estimating the average living area in homes in Ames. Because the sample mean is an unbiased estimator, the sampling distribution is centered at the true average living area of the the population, and the spread of the distribution indicates how much variability is induced by sampling only 50 home sales.

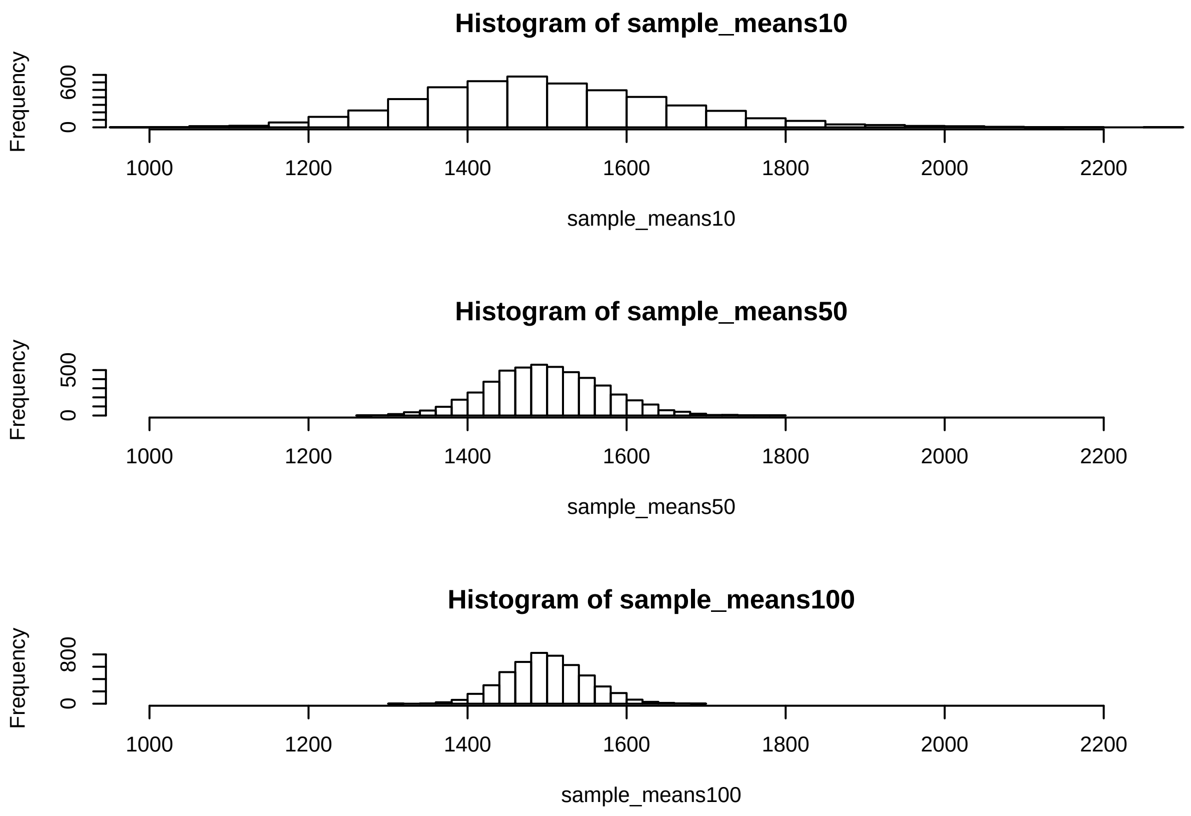
To get a sense of the effect that sample size has on our distribution, let's build up two more sampling distributions: one based on a sample size of 10 and another based on a sample size of 100.

```r
sample_means10 <- rep(NA, 5000)
sample_means100 <- rep(NA, 5000)

for(i in 1:5000){
  samp <- sample(area, 10)
  sample_means10[i] <- mean(samp)
  samp <- sample(area, 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.

```r
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 (to return to the default setting of plotting one at a time, use `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.

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

As the sample size increases, the center continues to cluster around the true population mean, however, the spread is reduced since the standard deviation is smaller. Therefore, this indicates that there is less variation between the sample mean and population mean as the sample size increases from 10 to 50 to 100 samples to calculate each sample mean.

On your own

So far, we have only focused on estimating the mean living area in homes in Ames. Now you'll try to estimate the mean home price.

- Take a random sample of size 50 from `price`. Using this sample, what is your best point estimate of the population mean?

```r
price_50sample <- sample(price, 50)

mean(price_50sample)
```

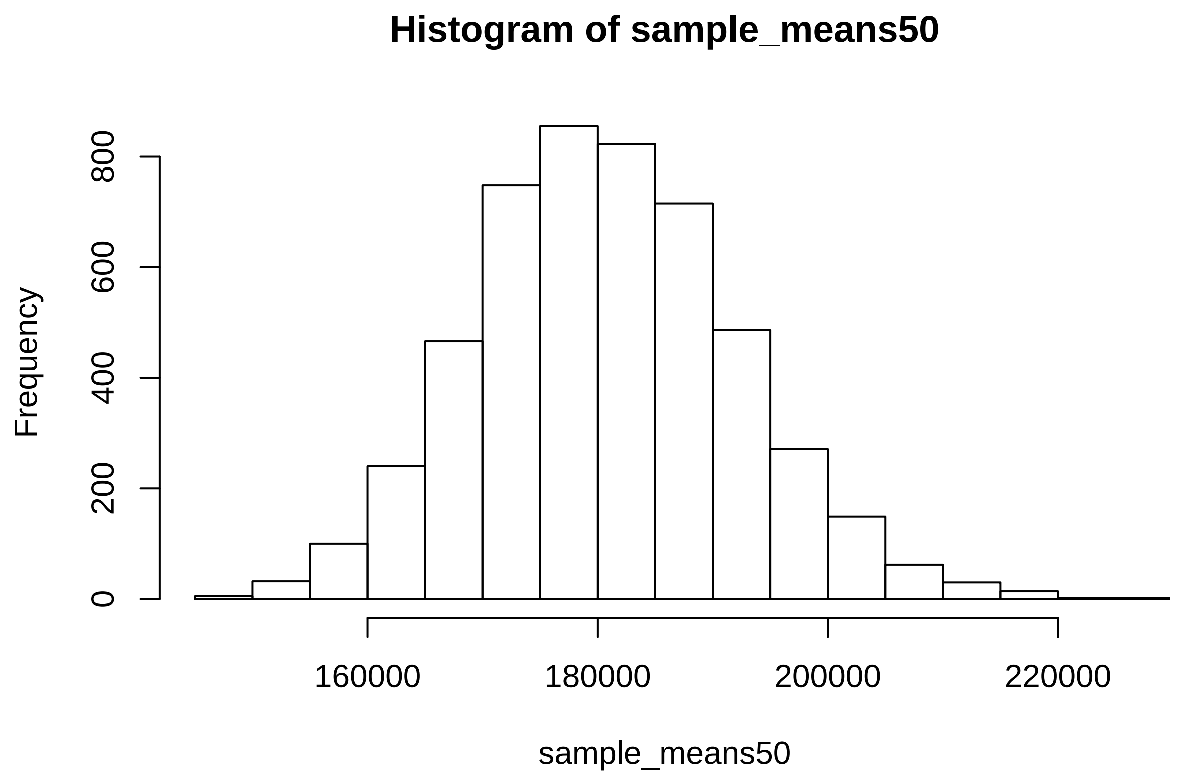
```
## [1] 188196.5
```

My best point estimate of the population mean from this random sample of size 50 for price is 188,196 dollars.

- Since you have access to the population, simulate the sampling distribution for $\bar{x}_{price}$ 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 home price of the population to be? Finally, calculate and report the population mean.

We will set this up similar to the R code we used to create samples for the mean living area in homes in Ames. Based on the plot below and the sampling distribution, it looks like the mean home price of the population will be around $180,000.

```r
# here's the for loop to compute the 5000 sample means
sample_means50 <- rep(NA, 5000)

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

# here's the plot of the sample means
hist(sample_means50, breaks = 20, xlim = range(sample_means50))
```



We can see that we were pretty close with our estimation of the population mean by plotting the sample means. The actual mean price of the population is 180,796 dollars, which is close to our estimation of 180,766 dollars.

```r
paste('sample_means50 mean value: ', mean(sample_means50))
```

```
## [1] "sample_means50 mean value:  180766.892396"
```

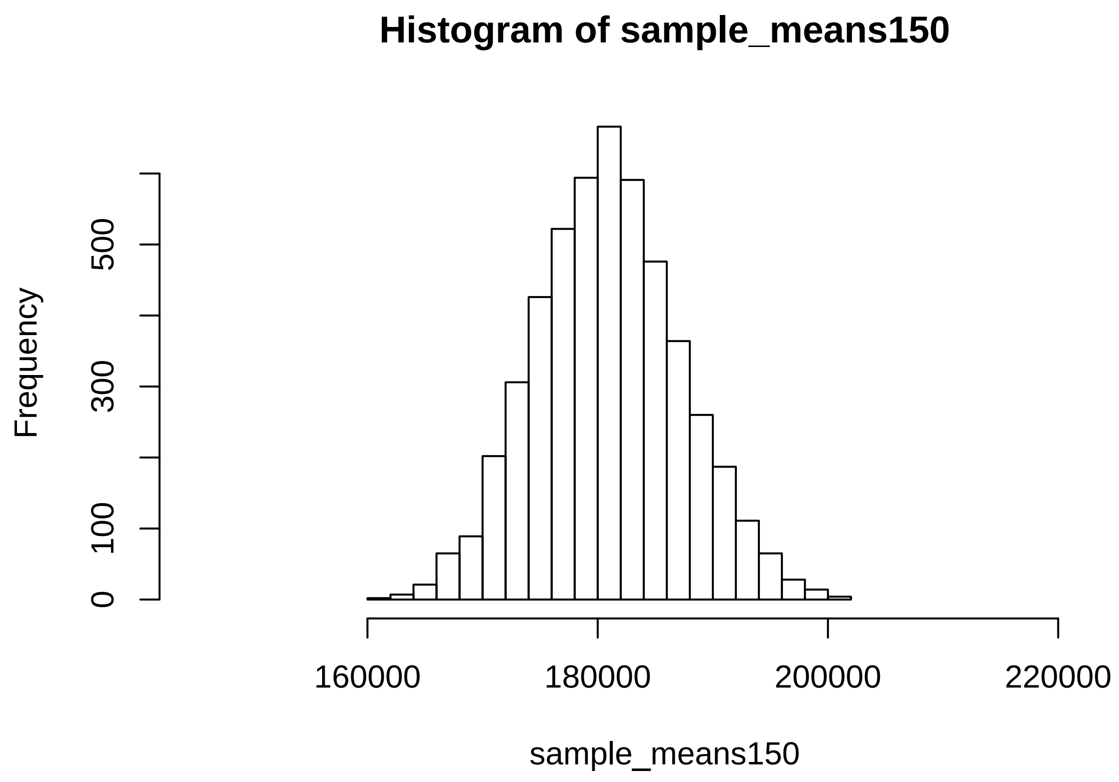
```r
# and to check, we can look at the actual population mean
paste('actual population mean: ', mean(price))
```

```
## [1] "actual population mean:  180796.060068259"
```

- Change your sample size from 50 to 150, then compute the sampling distribution using the same method as above, and store these means in a new vector called `sample_means150`. 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 sale price of homes in Ames?

```r
# here's the for loop to compute the 5000 sample means, adjusting the sample size
# to 150 from 50
sample_means150 <- rep(NA, 5000)

for(i in 1:5000){
   samp <- sample(price, 150)
```

```
    sample_means150[i] <- mean(samp)
}

# here's the plot of the sample means
hist(sample_means150, breaks = 20, xlim = range(sample_means50))
```



The shape of this sampling distribution is also close to normal, with a center just over 180,000 dollars. There is a much smaller spread for this sampling distribution compared to the sampling distribution for a sample size of 50. Additionally, the population mean seems to be more representative in this sample, as most sample means tend to hover slightly above 180,000 dollars relative to the sampling distribution for a sample size of 50, which had most sample means hover slightly below 180,000 dollars.

- 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?

The sampling distribution in 3 has a smaller spread (the distribution with a larger sample size). If we were concerned with making estimates that are more often close to the true value, we would prefer a distribution with a small spread and larger sample size.