HW3-Darshan Patel-3:30-5:30PM

Darshan Patel

10/18/2018

In the beginning of the 17th century, John Graunt wanted to determine the effect of the plague ont he population of England; two hundred years later, Pierre-Simon Laplace wanted to estimate the population of France. Both Graunt and Laplace implemented what is now called the `capture-recapture method.` This technique is used to not only count human populations (such as the homeless) but also animals in the wild.

In its simplest form, n_1 individuals are "captured," "tagged,"" and released. A while later, n_2 individuals are "captured" and the number of "tagged" individuals, m_2 is counted. If N is the true total population size, we can estimate it with $\hat{N}_{LP}$ as follows:

$$\hat{N}_{LP} = \frac{n_1 n_2}{m_2}$$

using the relation

$$\frac{n_1}{N} = \frac{m_2}{n_2}$$

This is called the Lincoln-Peterson estimator.

We make several strong assumptions when we use this method: (a) each individual is independently captured, (b) each individual is equally likely to be captured, (c) there are no births, deaths, immigration, or emigration of individuals (i.e., a closed population), and (d) the tags do not wear off (if it is a physical mark) and no tag goes unnoticed by a researcher.

Goal: In this assignment, you will develop a Monte-Carlo simulation of the capture-recapture method and investigate the statistical properties of the Lincoln-Peterson and Chapman estimators of population size N. (Since you are simulating your own data, you know the true value of the population size N allowing you to study how well these estimators work.)

```
# Import libraries
library(tidyverse)
```

```
## ── Attaching packages
─────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────── tidyverse 1.2.1
──
```

```
## ✔ggplot2 2.2.1     ✔purrr   0.2.5
## ✔tibble  1.4.2     ✔dplyr   0.7.6
```

```
## ✔tidyr   0.8.1     ✔stringr 1.3.0
## ✔readr   1.1.1     ✔forcats 0.3.0

## Warning: package 'tidyr' was built under R version 3.4.4

## Warning: package 'purrr' was built under R version 3.4.4

## Warning: package 'dplyr' was built under R version 3.4.4

## ── Conflicts
─────────────────────────────────────────────────────────────────────
─────────────────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────

tidyverse_conflicts() ──
## ✖dplyr::filter() masks stats::filter()
## ✖dplyr::lag()    masks stats::lag()
```

Question 1:

Simulate the capture-recapture method for a population of size $N = 5{,}000$ when $n_1 = 100$ and $n_2 = 100$ using the `sample()` function (assume that each individual is equally likely to be "captured"). Determine m_2 and calculate $\hat{N}_{LP}$.

Answer:

```
# Create variables
n <- 5000
n_1 <- 100
n_2 <- 100

# Sample the captured points
cap_one <- sample(1:n, n_1, replace = FALSE)
cap_two <- sample(1:n, n_2, replace = FALSE)

# Calculate m_2 and N_LP
m_2 <- length(intersect(cap_one, cap_two))
N_LP <- n_1 * n_2 / m_2
paste("m_2:", m_2, sep = ' ')

## [1] "m_2: 1"

paste("N_LP:", N_LP, sep = ' ')

## [1] "N_LP: 10000"
```

Question 2:

Write a function to simulate the capture-recapture procedure using the inputs: $N, n_1. n_2$ and the number of simulation runs. The function should output in list form (a) a data frame with two columns: the values of m_2 and $\hat{N}_{LP}$ for each iteration and (b) N. Run your simulation for

1,000 iterations for a population of size $N = 5{,}000$ where $n_1 = n_2 = 100$ and make a histogram of the resulting $\hat{N}_{LP}$ vector. Indicate N on the plot.

Answer:

```
# Function to simulate the capture-recapture procedure using the inputs: N,
n_1, n_2 and number of simulation runs
capture_recapture = function(N, n_1, n_2, runs){

  # Create vector to store m_2s and N_LPs
  m_2 <- c()
  N_LP <- c()

  # For each run, perform the capture-recapture procedure
  for(i in 1:runs){

    # Sample the captured points
    cap_one <- sample(1:N, n_1, replace = FALSE)
    cap_two <- sample(1:N, n_2, replace = FALSE)

    # Finds the points that are recaptured
    recaptured <- length(intersect(cap_one, cap_two))

    # Append to m_2 vector
    m_2 <- c(m_2, recaptured)

    # Calculate N_LP and append to its vector
    N_LP <- c(N_LP, (n_1 * n_2 / recaptured))
  }

  # Return a list where the first element is a dataframe of m_2 and N_LP
  # and the second element is the actual N
  return(list(df = data.frame(m_2, N_LP), N = N))
}
```

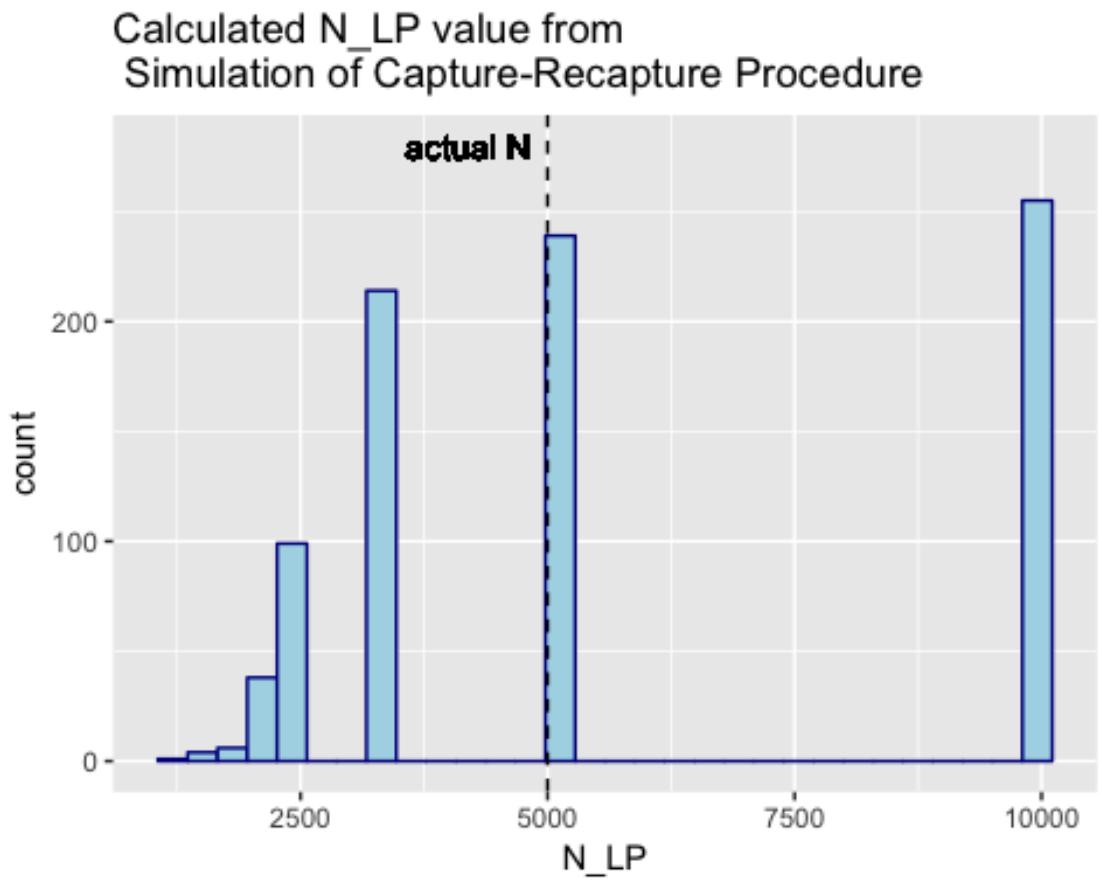
Simulation:

```
# Perform a simulation using 1000 runs
simulation <- capture_recapture(5000, 100, 100, 1000)
df <- simulation$df

# Plot a histogram of the calculated N_LP values
ggplot(df, aes(x = df$N_LP)) + geom_histogram(color = "darkblue", fill =
"lightblue") + ggtitle("Calculated N_LP value from \n Simulation of Capture-
Recapture Procedure") + geom_vline(aes(xintercept = 5000), color = "black",
linetype = "dashed") + geom_text(aes(x = simulation$N - 800, y = 280, label =
"actual N")) + labs(x = "N_LP")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 144 rows containing non-finite values (stat_bin).
```



Question 3:

What percent of the estimated population values in Question 2 were infinite? Why can this occur?

Answer:

```
per_inf <- paste(100 * length(which(is.infinite(simulation$df$N_LP))) /
nrow(simulation$df), "%", sep = ' ')
per_inf

## [1] "14.4 %"
```

The percentage of infinite N_{LP} calculated is 14.4 %. This occurs because in some rare cases, there was no recapture of any points, meaning $m_2 = 0$. Thus in the calculation of N_{LP}, there is a 0 in the denominator, causing R to give an infinite result.

Question 4:

An alternative to the Lincoln-Peterson estimator is the Chapman estimator:

$$\hat{N}_C = \frac{(n_1 + 1)(n_2 + 1)}{m_2 + 1} - 1$$

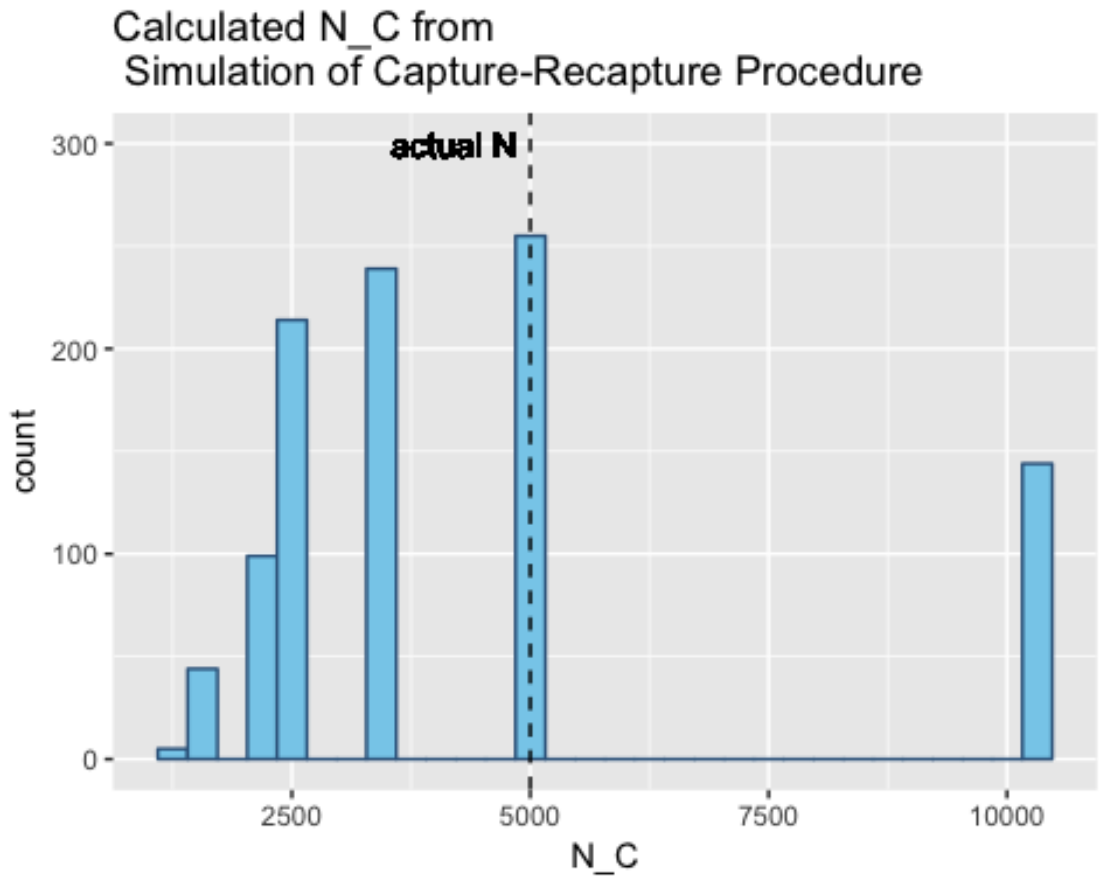
Use the saved m_2 values from question 2 to compute the corresponding Chapman estimates for each iteration of your simulation. Construct a histogram of the resulting $\hat{N}_C$ estimates, indicating N on the plot.

Answer:

```
# Calculate N_C from the m_2 values from the previous simulation runs
N_C <- data.frame(chapman = (((n_1 + 1)*(n_2 + 1))/(simulation$df$m_2 + 1)) -
1)

# Plot a histogram of the calculated N_C values
ggplot(N_C, aes(x = N_C$chapman)) + geom_histogram(color = "steelblue4", fill
= "skyblue") + ggtitle("Calculated N_C from \n Simulation of Capture-
Recapture Procedure") + geom_vline(aes(xintercept = 5000), color = "black",
linetype = "dashed") + geom_text(aes(x = simulation$N - 800, y = 300, label =
"actual N")) + labs(x = "N_C")

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Question 5:

An estimator is considered *unbiased* if, on average, the estimator equals the true population value. For example, the sample mean $\overline{x} = \sum_{i=1}^{n} x_i / n$ is unbiased because on average the sample mean $\overline{x}$ equals the population mean μ (i.e., the sampling distribution is centered

around μ). This is a desirable property for an estimator to have because it means our estimator is not systematically wrong. To show that an estimator $\hat{\theta}$ is an unbiased estimator of the true value θ, we would need to mathematically prove that $\mathrm{E}[\hat{\theta}] - \theta = 0$ where $\mathrm{E}[\cdot]$ is the expectation (i.e., theoretical average). Instead, we will investigate this property empirically by replacing the theoretical average $\mathrm{E}[\hat{\theta}]$ with the sample average of the $\hat{\theta}$ values from our simulation (i.e., $\sum_{i=1}^{n_{\text{sim}}} \hat{\theta} \, / n_{\text{sim}}$ where n_{sim} is the number of simulation runs; θ is N in this case and $\hat{\theta}$ is either $\hat{N}_{LP}$ or $\hat{N}_{C}$ as both are ways to estimate N).

Estimate the bias of the Lincoln-Peterson and Chapman estimators based on the results of your simulation. Is either estimator unbiased when $n_1, n_2 = 100$?

Answer: Note that to compute the mean of $N_L P$, the infinites should be ignored and n_{sim} should be properly instated.

```
# Isolate the finite N_LP values
N_LP_finite <- simulation$df$N_LP[is.finite(simulation$df$N_LP)]

# Calculate the bias of N_LP
bias_LP <- mean(N_LP_finite) / length(N_LP_finite)

# Calculate the bias of N_C
bias_chapman <- mean(N_C[,1]) / length(N_C[,1])
paste("The bias of the Lincoln-Peterson estimator is", bias_LP, sep = ' ')

## [1] "The bias of the Lincoln-Peterson estimator is 6.54914951503354"

paste("The bias of the Chapman estimator is", bias_chapman, sep = ' ')

## [1] "The bias of the Chapman estimator is 4.40856845873016"
```

Neither of the estimators are unbiased when $n_1, n_2 = 100$.

Question 6:

Based on your findings, is the Lincoln-Peterson or Chapman estimator better? Explain your answer.

Answer: Based on the findings, the Chapman estimator does better to estimate N than the Lincoln-Peterson estimator. One reason is that the bias of the estimator calculated was lower than the one calculated using the Lincoln-Peterson estimator. Another reason is that the Lincoln-Peterson estimator was accustomed to calculating infinities which is theoretically not possible in real life scenarios.

Question 7:

Explain why the assumptions (a), (b) and (c) used to make the Lincoln-Peterson estimator are unrealisitic.

Answer: Unrealistic Assumptions:

- A: When one individual is captured, it is possible that another individual is captured in a nearby vicinity as well
- B: Each individual may have factors that influence its chances of being captured, such as health or agility
- C: Births and deaths events cannot be held constant in the real world as they are natural events.