Chak Wong Computation and Statistics

Homework 3

1. Simulate the capture-recapture method for a population of size N = 5, 000 when n1 = 100 and n2 = 100 using the sample() function (we assume that each individual is equally likely to be "captured"). Determine m2 and calculate N^LP using Eq.1.

Answer: For this sample, m2 is 1 and N^LP is 10000. However, two things should be noted: 1) m2 fluctuates according to common elements between the first two catches 2) N^LP calculates the ratio between the first two catches and the intersection between the first two catches to estimate the true population size. Thus, m2 and N^LP is entirely random.

# Set initial parameters

N <- 5000

n.one <- 100

n.two <- 100

# Use sample() function to simulate first catch

catch.one <- sample(N, n.one, replace=T, prob=NULL)

# Use sample() function to simulate second catch

catch.two <- sample(N, n.two, replace=T, prob=NULL)

# Find common elements between catch.one and catch.two

m.two <- intersect(catch.one, catch.two)

# Amount of common elements

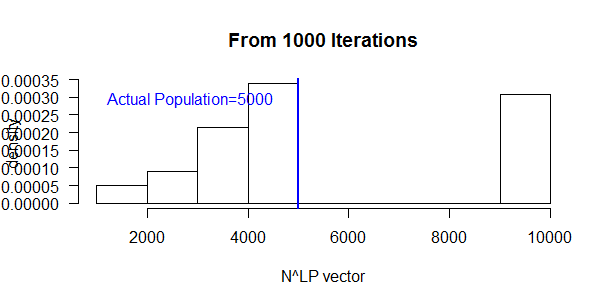
m.two <- length(m.two)

# Calculate N^LP

N.hat.LP <- (n.one\*n.two)/(m.two)

2. Write a function to simulate the capture-recapture procedure using the inputs: N, n1, n2, and the number of simulation runs. The function should output in list form (a) a data frame with two columns: the values of m2 and N^LP for each iteration and (b) N. Run your simulation for 1,000

iterations for a population of size N =5,000 and make a histogram of the resulting N^LP vector. Indicate N on your plot.



# Motivate capture-recapture () function

cap.recap <- function(n.sim){

#n.sim: number of iterations

#list.sim.m.two: m2 values for entire simulation

#list.sim.N.hat.LP: N.hat.LP values for entire simulation

list.sim.m.two <- NULL

list.sim.N.hat.LP <- NULL

# Obtain m2 & N^LP for each iteration

for(i in 1:n.sim){

# First catch

sim.catch.one <- sample(N, n.one, replace=T, prob=NULL)

sim.catch.one <- as.numeric(sim.catch.one)

# Second catch

sim.catch.two <- sample(N, n.two, replace=T, prob=NULL)

sim.catch.two <- as.numeric(sim.catch.two)

# Common elements between first and second catches

sim.m.two <- intersect(sim.catch.one, sim.catch.two)

# if m2 == 0

if(length(sim.m.two) == 0){

list.sim.m.two[i] <- as.character(0)

}else{

sim.m.two.values <- sim.m.two

sim.m.two.values <- toString(sim.m.two.values)

list.sim.m.two[i] <- sim.m.two.values

}# end if/else

# Calculate N^LP

sim.m.two <- length(sim.m.two)

sim.N.hat.LP <- (n.one\*n.two)/(sim.m.two)

list.sim.N.hat.LP[i] <- sim.N.hat.LP

}# end m2 & N^LP

# Combine list of m2 & N^LP values into data frame

list.sim.m.two.N.hat.LP <- data.frame(list.sim.m.two, list.sim.N.hat.LP)

# Return output of m2 & N^LP data frame and N

return(list("m2 & N^LP"=list.sim.m.two.N.hat.LP, "POP"=N))

}# end cap.recap() function

# Present Output

Output<-cap.recap(1000)

# Extract N^LP vector from Output

Output <- data.frame(Output)

colnames(Output) <- c("m2", "NLP", "POP")

v.NLP <- Output[,2]

# Plot histogram

hist(v.NLP, freq=FALSE, las=TRUE,

main=paste("From 1000 Iterations"),

xlab="N^LP vector", ylab="density")

legend("topleft", legend="Actual Population=5000", text.col="blue",bty="n")

abline(v = N, col = "blue", lwd = 2)

# pdf("HW3\_Question2.pdf", height=8, width=12)

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

prop.table(table(Output$NLP))

1250 1666.66666666667 2000 2500 3333.33333333333

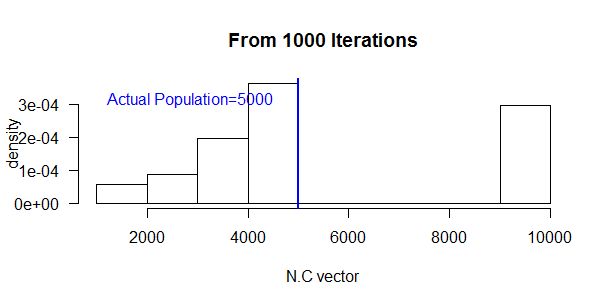
0.001 0.010 0.027 0.086 0.175

5000 10000 Inf

0.290 0.276 0.135

Answer: The estimated population values in question 2 that were infinite was 14.7% of the total number of values. This occurs when the two collected samples fail to have a common element (i.e. fish), which results in the Lincoln-Peterson formula's denominator to be 0 and the subsequent answer be infinite.

4. An alternative to the Lincoln-Peterson estimator is the Chapman estimator: NˆC=(((n1+1)(n2+1))/(m2+1)−1. Use the saved m2 values from question 2 to compute the Chapman estimate for each case. Construct a histogram of the resulting NˆC estimates, indicating N on your plot.



# Extract m2 values from Question 2 Output

v.m.two <- Output[,1]

v.m.two <- data.frame(v.m.two)

s <- strsplit(as.character(v.m.two[,1]), ", ")

v.m.two <- sapply(s, length)

# For Loop to calculate Chapman estimator

N.C <- NA

for(i in 1:length(v.m.two)){

N.C <- ((n.one+1)\*(n.two+1)/(v.m.two+1))-1

N.C[i] <- list(N.C)

} # end For Loop

# Construct Histrogram and plot actual population

hist(as.numeric(N.C), freq=FALSE, las=TRUE,main=paste("From 1000 Iterations"),

xlab="N.C vector", ylab="density")

legend("topleft", legend="Actual Population=5000", text.col="blue",bty="n")

abline(v = N, col = "blue", lwd = 2)

# pdf("HW3\_Question2.pdf", height=8, width=12)

5. Estimate the bias of the Lincoln-Peterson and Chapman estimators based on the results of your simulation. Is either estimator unbiased when n1, n2 = 100?

Answer: When n1,n2=100, neither estimator were found to be unbiased - specifically, the Lincoln-Peterson and Chapman estimators' values were -91.07143 and -1214.831, respectively or were ~91 and ~1215, respectively, away from the true population (n=5000). However, it should be noted that 1) these values are contingent upon how many values were sampled from catches one and two, the overall population, and simply the luck of the draw, and 2) mathematically, the Chapman estimator is built to prevent a 0 denominator, thus and infinity output as a result.

# Calculate N^LP estimator's biasedness

sum.NLP <-sum(as.numeric(v.NLP), na.rm=TRUE)

mean.NLP <- sum.NLP/1000

biased.NLP <- mean.NLP-N

# Calculate NC estimator's biasedness

sum.NC <- sum(unlist(N.C), na.rm=TRUE)

mean.NC <- sum.NC/1000

biased.NC <- mean.NC-N

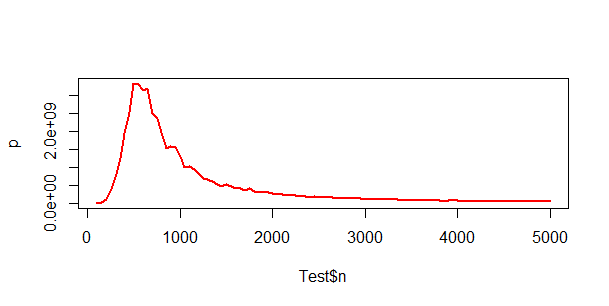
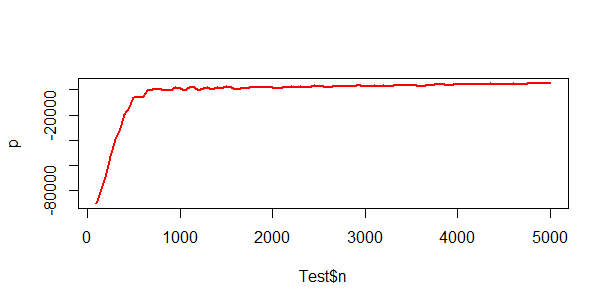
biased.NLP

biased.NC

6. Based on your findings thus far, is the Lincoln-Peterson or Chapman estimator better? Justify your answer.

Answer: My current findings suggests that the Chapman estimator is better than the Lincoln-Peterson estimator. As mentioned in Question 5's answers, L-P estimator(n=~4901) is ~91 away from the true population (n=5000), while the Chapman estimator(n=~3785) is ~1215 away from the true population; also, these values are contingent upon how many and what values were sampled from catches one and two, the overall population, and simply the luck of the draw. Theoretically, the mathematical formula for the Chapman Estimator forces the estimator to decrease dramatically as the amount of fishes caught in both catches increases in comparison to the Lincoln-Peterson estimator, but it also prevents a 0 denominator.

7. Can we get better estimates using larger sample sizes? Let’s restrict n = n1 = n2. Write a function which computes the bias and variance of the Chapman estimator for varying sample sizes n. The inputs for this function should be: N, number of simulation runs per sample size, and n. The function should return in list form (a) a data frame with three columns: n, bias of estimator, and standard deviation of estimator for each sample size and (b) the true population size. Run this function using the following arguments: (a) N =100,000 for each sample size, (b) 1,000 simulation runs, and (c) n ranging from 100 to 5,000 (use seq(from=100, to=5000, by=50)). Include in your function an indicator of your progress every time you complete runs for 10 sample sizes. Based on your results, construct two plots: (a) bias versus n and (b) variance versus n (e.g., y-axis variable vs. x-axis variable). Indicate zero on both plots and connect the points with a line. Describe what you see.



Answer: There is a positive, curvelinear relationship between sample size and bias of estimator - that is, as increasing sample size results in the estimator to become unbiased, but it quickly becomes biased again. There is a positive, linear relationship between sample size and variance of estimator. It makes sense because we squared a value when we converted standard deviation to variance.

# Population Size

N <- 100000

# Iterations per Sample Size

n.sim <- 1000

# Various Sample Sizes

samples.sim <- seq(from=100, to=5000, by=50)

cap.recap.partdeux <- function(N, n.sim, samples.sim){

n <- NA # sample size

bias.of.est <- NA # bias of estimator

sd.of.est <- NA # standard deviation of estimators by sample

l.sim.chap <- NA # temporary list of estimators to use for later

# For Loop: Various Sample Sizes

for(i in 1:length(samples.sim)){

# For Loop: Iterations per Sample Size

for (j in 1:n.sim){

# Catch One

sim.one <- sample(N, samples.sim[i], replace=T, prob=NULL)

# Catch Two

sim.two <- sample(N, samples.sim[i], replace=T, prob=NULL)

# Find Common Elements

sim.m.two <- intersect(sim.one, sim.two)

# Amount of Common Elements

sim.l.m.two <- length(sim.m.two)

# Calculate Chapman Estimator

sim.chap <- ((samples.sim[i]+1)\*(samples.sim[i}+1)/(sim.l.m.two+1))-1

l.sim.chap[j] <- list(sim.chap)

} # End For Loop: Iterations per Sample Size

# Calculate bias of estimator for each sample

sum.b.est <- sum(unlist(l.sim.chap), na.rm=T)

bias.est <- (sum.b.est/n.sim)-N

bias.of.est[i] <- bias.est

# Calculate standard deviation of estimator for each sample

sd.est <- sd(unlist(l.sim.chap), na.rm = TRUE)

sd.of.est[i] <- sd.est

# Sample Size

n[i] <- samples.sim[i]

# Modulus operation

if(i %% 10==0) {

# Print on the screen some message

cat(paste0("iteration: ", i, "\n"))

}

} # End For Loop: Various Sample Sizes

# Return Three Columns and make Data Frame

Three <- (data.frame(n, bias.of.est, sd.of.est))

# List of Data Frame with True Population

Output <- (list(Three, "POP"=N))

return(Output)

} # End Function

Output <- cap.recap.partdeux(N=100000, n.sim=1000, samples.sim)

Test <- data.frame(Output)

# Plot bias of estimator on sample size

# Regular Plot

plot(Test$'n', Test$'bias.of.est')

# Exponential Probability Density Function

y=dexp(Test$"n",rate=1)

plot(Test$"n",Test$bias.of.est, type="l",lwd=2,col="red",ylab="p")

# Plot variance of estimator on sample size

# Regular Plot

var.of.est <- (Test$'sd.of.est')^2

plot(Test$'n', var.of.est)

# Exponential Probability Density Function

y=dexp(Test$"n",rate=1)

plot(Test$"n",var.of.est, type="l",lwd=2,col="red",ylab="p")

8. An estimator ˆθ is called consistent if, as the sample size goes to infinity, the bias of the estimator goes to zero. That is, E[ˆθ] gets closer to θ as the sample size increases.This is another desirable property for an estimator. For the simpler case where n1 = n2, could NˆC be a consistent estimator? Justify your answer.

Answer: The Chapman Estimator is a consistent estimator of the true population for two reasons. First, as mentioned in Question 6, the Chapman estimator is that when there isn't an overlap of common elements, the denominator is articifically set to 1 to prevent the quotient to be INF. Second, as shown in Question 7, the bias and variance of the chapman estimator

# decreases exponentially as the sample size decreases.

9. Explain why the assumptions (a), (b), and (c) listed on the first page are unrealistic.

Answer: Assumptions (a), (b), and (c) listed on the first page are unrealistic condition inapplicable in a real-life, naturalistic setting. Assumption (a) presumes the two catches, and each fish being caught, are independent of the others, but location, types of fish, time of catches, etc., play a role in how and when fishes are being caught. Assumption (b) presumes that each

individual is equally likely to be captured, but it's unrealistic because the fisherman(you) simply has access to catching fishes within his vicinity not throughout the entire body of water simultaneously. Assumption (c) presumes the fish population remains a closed population, whereby there are no births, deaths, immigration, or emigration of individuals; but it's also unrealistic because any of the aforementioned conditions/scenarios are intrinsic to life

and what's supposed to occur in nature.