Simple Random Sampling

Longhai Li

Table of Contents

library(latex2exp)

# A function for doing data analysis for srs sample

#   
# sdata -- a vector of sampling survey data  
# N -- population size   
srs\_mean\_est <- function (sdata, N)  
{  
 n <- length (sdata)  
 ybar <- mean (sdata)  
 se.ybar <- sqrt((1 - n / N)) \* sd (sdata) / sqrt(n)   
 mem <- qt (0.975, df = n - 1) \* se.ybar  
 c (ybar = ybar, se = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem)  
}

# Analysis of agsrs.csv Data

## step by step calculation without using a function

## read survey data  
agsrs <- read.csv ("data/agsrs.csv")  
head(agsrs)

## county state acres92 acres87 acres82 farms92 farms87 farms82  
## 1 COFFEE COUNTY AL 175209 179311 194509 760 842 944  
## 2 COLBERT COUNTY AL 138135 145104 161360 488 563 686  
## 3 LAMAR COUNTY AL 56102 59861 72334 299 362 447  
## 4 MARENGO COUNTY AL 199117 220526 231207 434 471 622  
## 5 MARION COUNTY AL 89228 105586 113618 566 658 748  
## 6 TUSCALOOSA COUNTY AL 96194 120542 134616 436 521 650  
## largef92 largef87 largef82 smallf92 smallf87 smallf82 region  
## 1 29 28 21 57 47 66 S  
## 2 37 41 42 12 44 47 S  
## 3 4 4 3 16 20 30 S  
## 4 48 66 62 14 11 28 S  
## 5 7 9 9 11 23 27 S  
## 6 20 17 23 18 32 29 S

## extract the variable of interest  
sdata <- agsrs$acres92  
N <- 3078  
  
## do calculation  
n <- length (sdata)  
ybar <- mean (sdata)  
se.ybar <- sqrt((1 - n / N)) \* sd (sdata) / sqrt(n)   
mem <- qt (0.975, df = n - 1) \* se.ybar  
## return estimate vector for pop mean  
c (Est. = ybar, S.E. = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem)

## Est. S.E. ci.low ci.upp   
## 297897.05 18898.43 260706.26 335087.84

## return estimate vector for pop total  
c (Est. = ybar, S.E. = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem) \* N

## Est. S.E. ci.low ci.upp   
## 916927110 58169381 802453859 1031400361

## Write a function for repeated use

## sdata --- a vector of original survey data  
## N --- population size  
## to find total, multiply N to the estimate returned by this function  
srs\_mean\_est <- function (sdata, N = Inf)  
{  
 n <- length (sdata)  
 ybar <- mean (sdata)  
 se.ybar <- sqrt((1 - n / N)) \* sd (sdata) / sqrt(n)   
 mem <- qt (0.975, df = n - 1) \* se.ybar  
 c (Est. = ybar, S.E. = se.ybar, ci.low = ybar - mem, ci.upp = ybar + mem)  
}  
  
agsrs <- read.csv ("data/agsrs.csv")  
  
# inference for mean of variable acres92  
srs\_mean\_est (agsrs[,"acres92"], N = 3078)

## Est. S.E. ci.low ci.upp   
## 297897.05 18898.43 260706.26 335087.84

# inference for total of variable acres92  
srs\_mean\_est (agsrs[,"acres92"], N = 3078) \* 3078

## Est. S.E. ci.low ci.upp   
## 916927110 58169381 802453859 1031400361

## Estimate the proportion of counties with fewer than 200K acres for farming in 1992

acres92.is.fewer.200k <- as.numeric (agsrs[,"acres92"] < 200000)  
srs\_mean\_est (acres92.is.fewer.200k, N = 3078)

## Est. S.E. ci.low ci.upp   
## 0.51000000 0.02746498 0.45595084 0.56404916

# inference for total number of counties with fewer than 200K acres for farming in 1992  
srs\_mean\_est (acres92.is.fewer.200k, N = 3078) \* 3078

## Est. S.E. ci.low ci.upp   
## 1569.78000 84.53722 1403.41670 1736.14330

## Comparing with true value

agpop <- read.csv ("data/agpop.csv", na = "-99")  
#true mean  
mean (agpop[, "acres92"], na.rm = T)

## [1] 308582.4

# true total  
sum (agpop[, "acres92"], na.rm = T)

## [1] 943953599

# true proportion of counties with less than 200K acres for farming  
mean (agpop[, "acres92"] < 200000, na.rm = T)

## [1] 0.5145472

# true number of counties with less than 200K acres for farming  
sum (agpop[, "acres92"] < 200000, na.rm = T)

## [1] 1574

# A Simulation Demonstration of SRS Inference

# read population data   
agpop <- read.csv ("data/agpop.csv")  
## remove those counties with na  
agpop <- subset( agpop, acres92 != -99)

## Simulation

###################### working with variable 'acres92' ##################  
# sample size  
n <- 300  
# population size  
N <- nrow (agpop)  
  
# true value of population mean  
ybarU <- mean (agpop[,"acres92"]); ybarU

## [1] 308582.4

# true value of deviation of sample mean  
true.se.ybar <- sqrt (1- n/N) \* sd (agpop[,"acres92"]) / sqrt (n); true.se.ybar

## [1] 23320.29

## one srs sampling  
# srs sampling  
srs <- sample (1:N,n); srs

## [1] 2033 450 1815 342 2043 2440 196 2318 1664 786 934 1833 2443 1188 2499  
## [16] 2435 987 2986 2640 2611 1082 1276 1039 2199 2592 562 1147 317 1107 40  
## [31] 1950 2015 81 1873 810 1397 2579 2412 517 1742 2639 1929 2075 1963 1916  
## [46] 1264 2521 1924 2514 1962 1567 1585 742 545 569 1116 1192 478 608 388  
## [61] 1666 120 2607 2336 2493 908 1177 575 486 1904 1509 1075 2381 898 1681  
## [76] 2084 935 1596 87 2421 373 2677 2471 737 2418 132 729 1170 11 97  
## [91] 263 3032 194 1274 2959 257 713 2095 2253 83 1623 2462 2209 1547 414  
## [106] 1388 2937 1159 1093 3041 697 275 1790 2338 3 698 2562 801 46 2002  
## [121] 2243 400 351 1920 2274 2416 1760 1513 2010 93 2746 1325 2260 966 1689  
## [136] 3015 3042 1446 363 1821 382 168 1251 612 322 924 1400 66 2975 80  
## [151] 624 2137 1195 51 1356 1957 455 1905 1103 461 814 304 1005 2133 2549  
## [166] 2388 1727 2469 1961 2628 1440 449 2070 1377 1442 1 65 2622 759 1098  
## [181] 1396 1468 2192 425 2252 70 1770 1168 724 493 1740 1979 1734 1636 277  
## [196] 892 1697 2423 312 2732 896 962 1189 2522 749 746 434 687 647 237  
## [211] 1945 1113 2524 210 357 1154 622 1000 2982 2474 2712 1031 2473 1363 2065  
## [226] 269 1071 2097 2480 315 430 1805 181 2770 1723 948 91 895 1602 184  
## [241] 2442 377 1614 1467 1230 1327 1890 1702 1096 1610 1180 864 375 1025 1611  
## [256] 1095 1981 1898 1826 1453 2978 2518 441 1551 2377 632 1432 2641 1234 922  
## [271] 2384 2726 942 1518 152 32 2063 285 198 323 2806 691 1716 1407 348  
## [286] 2118 1674 1369 609 2709 1150 2515 1629 439 799 1494 1144 530 1271 877

# get data of variable "acres92"  
sdata <- agpop [srs, "acres92"]; sd (sdata)

## [1] 326901.8

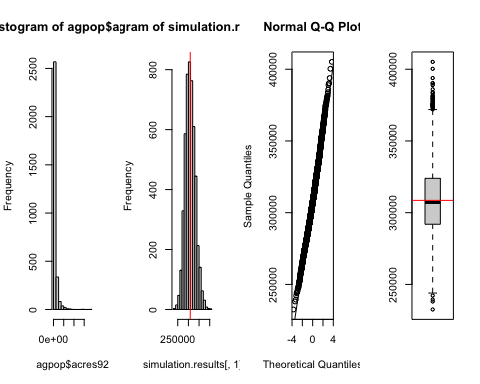
# analysis  
srs\_mean\_est (sdata, N)

## Est. S.E. ci.low ci.upp   
## 298689.42 17924.32 263415.61 333963.22

## repeated srs sampling  
nres <- 5000 # number of repeated sampling  
simulation.results <- matrix (0, nres, 4) # matrix recording repeated results  
colnames(simulation.results) <- c( "Est.", "S.E.", "ci.low", "ci.upp")  
  
for (i in 1:nres)  
{  
 srs <- sample (N, n)  
 sdata <- agpop [srs, "acres92"]  
 simulation.results [i,] <- srs\_mean\_est (sdata, N)  
}  
  
head(simulation.results)

## Est. S.E. ci.low ci.upp  
## [1,] 319712.2 22355.02 275719.1 363705.3  
## [2,] 296594.6 23847.78 249663.8 343525.3  
## [3,] 279454.5 20363.84 239379.9 319529.1  
## [4,] 292638.2 27075.43 239355.6 345920.7  
## [5,] 299742.8 27145.96 246321.5 353164.2  
## [6,] 329150.2 27225.44 275572.5 382728.0

# look at the distribution of sample mean  
par (mfrow= c(1,4))  
hist (agpop$acres92)  
hist (simulation.results[,1])  
abline (v = ybarU, col = "red")  
qqnorm (simulation.results[,1]); qqline(simulation.results[,1])  
boxplot (simulation.results[,1])  
abline (h = ybarU, col = "red")



mean (simulation.results[,1])

## [1] 308512.8

ybarU

## [1] 308582.4

sd (simulation.results [,1])

## [1] 23593.81

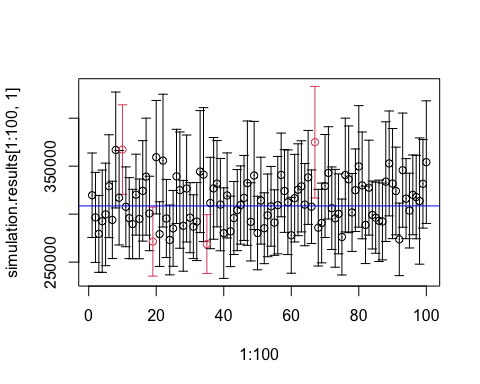
true.se.ybar

## [1] 23320.29

# look at coverage rate through simulation  
simulation.results <- cbind (simulation.results, (simulation.results[,3] < ybarU) \* (ybarU < simulation.results[,4] ))  
colnames(simulation.results)[5] <- "Covered?"  
head(simulation.results)

## Est. S.E. ci.low ci.upp Covered?  
## [1,] 319712.2 22355.02 275719.1 363705.3 1  
## [2,] 296594.6 23847.78 249663.8 343525.3 1  
## [3,] 279454.5 20363.84 239379.9 319529.1 1  
## [4,] 292638.2 27075.43 239355.6 345920.7 1  
## [5,] 299742.8 27145.96 246321.5 353164.2 1  
## [6,] 329150.2 27225.44 275572.5 382728.0 1

library("plotrix")   
par(mfrow=c(1,1))  
plotCI(x=1:100,  
 y=simulation.results[1:100,1],  
 li = simulation.results[1:100,3],  
 ui = simulation.results[1:100,4],  
 col = 2-simulation.results[,5])  
abline(h=ybarU, col = "blue")



# actual coverage rate in the simulation  
mean (simulation.results[,5])

## [1] 0.9338