# Sampling Design and Survey Practice Lab #2

TA - Seungkyu Kim

2022-10-12

### Install and load packages

```
name_pkg <- c("survey", "sampling", "SDAResources")
name_pkg <- unique(name_pkg)
bool_nopkg <- !name_pkg %in% rownames(installed.packages())
if (sum(bool_nopkg) > 0) {
  install.packages(name_pkg[bool_nopkg])
}
invisible(lapply(name_pkg, library, character.only = T))
```

## 1. 표본 배정 (Allocation Methods)

```
data(agpop)  # load the data set
names(agpop) # list the variable names
```

```
##  [1] "county"   "state"    "acres92"  "acres87"  "acres82"  "farms92"
##  [7] "farms87"  "farms82"  "largef92" "largef87" "largef82" "smallf92"
## [13] "smallf87" "smallf82" "region"
```

```
head(agpop)  # take a look at the first 6 obsns
```

```
##                 county state acres92 acres87 acres82 farms92 farms87 farms82
## 1 ALEUTIAN ISLANDS AREA    AK  683533  726596  764514      26      27      28
## 2        ANCHORAGE AREA    AK   47146   59297  256709     217     245     223
## 3        FAIRBANKS AREA    AK  141338  154913  204568     168     175     170
## 4           JUNEAU AREA    AK     210     214     127       8       8      12
## 5  KENAI PENINSULA AREA    AK   50810   85712   98035      93     119     137
## 6        AUTAUGA COUNTY    AL  107259  116050  145044     322     388     453
##   largef92 largef87 largef82 smallf92 smallf87 smallf82 region
## 1       14       16       20        6        4        1      W
## 2        9       10       11       41       52       38      W
## 3       25       28       21       12       18       25      W
## 4        0        0        0        5        4        8      W
## 5        9       18       17       12       18       19      W
## 6       25       32       32        8       19       17      S
```

```r
nrow(agpop)  # number of rows, 3078
```

```
## [1] 3078
```

```r
unique(agpop$region) # take a look at the four regions, NC, NE, S, W
```

```
## [1] "W"  "S"  "NE" "NC"
```

```r
table(agpop$region)  # number of counties in each stratum
```

```
##
##   NC   NE    S    W
## 1054  220 1382  422
```

### 비례 배정 (Proportional allocation)

```r
popsize <- table(agpop$region)
propalloc <- 300*popsize/sum(popsize)
propalloc
```

```
##
##       NC       NE        S        W
## 102.7290  21.4425 134.6979  41.1306
```

```r
# Round to nearest integer
propalloc_int <- round(propalloc)
propalloc_int
```

```
##
##  NC  NE   S   W
## 103  21 135  41
```

```r
sum(propalloc_int) # check that stratum sample sizes sum to 300
```

```
## [1] 300
```

### 네이만 배정 (Neyman allocation)

```r
stratvar <- c(1.1,0.8,1.0,2.0)
# Make sure the stratum variances in stratvar are in same
#  order as the table in popsize
neymanalloc <- 300*(popsize*sqrt(stratvar))/sum(popsize*sqrt(stratvar))
neymanalloc
```

```
##
##        NC        NE         S         W
## 101.07640  17.99204 126.36327  54.56828
```

```
neymanalloc_int <- round(neymanalloc)
neymanalloc_int
```

```
##
##  NC  NE   S   W
## 101  18 126  55
```

```
sum(neymanalloc_int)
```

```
## [1] 300
```

### 최적 배정 with 비용함수 (Optimal allocation)

```
relcost <- c(1.4,1.0,1.0,1.8)
# Make sure the relative costs in relcost are in same
# order as the table in popsize
optalloc <- 300*(popsize*sqrt(stratvar/relcost))/sum(popsize*sqrt(stratvar/relcost))
optalloc
```

```
##
##        NC        NE         S         W
##  94.75776  19.95766 140.16833  45.11626
```

```
optalloc_int <- round(optalloc)
optalloc_int
```

```
##
##  NC  NE   S   W
##  95  20 140  45
```

```
sum(optalloc_int)
```

```
## [1] 300
```

## 2. 모집단으로부터 층화임의표집하기 (Selecting a Stratified Random Sample)

### Use the sample function with each stratum

```
# Select an SRS without replacement from each region with proportional allocation
# with total size n=300
regionname <- c("NC","NE","S","W")
# Make sure sampsize has same ordering as regionname
sampsize <- c(103,21,135,41)
```

```
# Set the seed for random number generation
set.seed(108742)
index <- NULL
N <- nrow(agpop)
for (i in 1:length(sampsize)) {
  index <- c(index,sample((1:N)[agpop$region==regionname[i]],
                          size=sampsize[i],replace=F))
}
strsample<-agpop[index,]
# Check that we have the correct stratum sample sizes
table(strsample$region)
```

```
##
##  NC  NE   S   W
## 103  21 135  41
```

```
# Print the first six rows of the sample to see
strsample[1:6,]
```

```
##                       county state acres92 acres87 acres82 farms92 farms87
## 1316            ISANTI COUNTY    MN  131563  142998  153003     680     817
## 2034          DEFIANCE COUNTY    OH  196759  206905  210781     830     987
## 864           ATCHISON COUNTY    KS  245099  233619  234730     686     694
## 553         DES MOINES COUNTY    IA  192467  210843  224770     681     753
## 1738              DUNN COUNTY    ND 1352738 1358843 1397141     650     733
## 1325 LAKE OF THE WOODS COUNTY    MN  103665  118959  119296     176     222
##      farms82 largef92 largef87 largef82 smallf92 smallf87 smallf82 region
## 1316     947       18       14        8       14       26       34     NC
## 2034    1033       25       20       18       40       50       50     NC
## 864      768       55       42       41       48       48       65     NC
## 553      815       33       30       24       56       56       72     NC
## 1738     697      358      368      361       19       13       34     NC
## 1325     230       30       35       26        4        4        1     NC
```

### Use the strata function from the sampling package

```
# Sort the population by stratum
agpop2<-agpop[order(agpop$region),]
# Use the strata function to select the units for the sample
# Make sure size argument has same ordering as the stratification variable
index2<-strata(agpop2,stratanames=c("region"),size=c(103,21,135,41),
               method="srswor")
table(index2$region)  # look at number of counties selected within each region
```

```
##
##  NC  NE   S   W
## 103  21 135  41
```

```
head(index2)
```

```
##    region ID_unit       Prob Stratum
## 2      NC       2 0.09772296       1
## 9      NC       9 0.09772296       1
## 27     NC      27 0.09772296       1
## 36     NC      36 0.09772296       1
## 42     NC      42 0.09772296       1
## 43     NC      43 0.09772296       1
```

```
strsample2<-getdata(agpop2,index2) # extract the sample
head(strsample2)
```

```
##              county state acres92 acres87 acres82 farms92 farms87 farms82
## 526    ADAMS COUNTY    IA  239800  243607  254071     643     688     737
## 533   BREMER COUNTY    IA  236668  235086  250402    1058    1140    1287
## 551  DECATUR COUNTY    IA  261494  278714  300684     648     715     769
## 560  FREMONT COUNTY    IA  302352  308796  306786     596     719     771
## 566   HARDIN COUNTY    IA  332358  337990  355823     986    1065    1208
## 567 HARRISON COUNTY    IA  399155  387190  408601     919    1024    1192
##     largef92 largef87 largef82 smallf92 smallf87 smallf82 region ID_unit
## 526       38       32       21       40       50       33     NC       2
## 533       25       18       11       96      116      109     NC       9
## 551       52       54       56       20       34       37     NC      27
## 560       91       72       51       37       59       50     NC      36
## 566       56       36       42       90      115      132     NC      42
## 567       88       62       51       60       60       66     NC      43
##           Prob Stratum
## 526 0.09772296       1
## 533 0.09772296       1
## 551 0.09772296       1
## 560 0.09772296       1
## 566 0.09772296       1
## 567 0.09772296       1
```

```
# Calculate the sampling weights
# First check that no probabilities are 0
sum(strsample2$Prob<=0)
```

```
## [1] 0
```

```
strsample2$sampwt<-1/strsample2$Prob
# Check that the sampling weights sum to the population sizes for each stratum
tapply(strsample2$sampwt,strsample2$region,sum)
```

```
##   NC   NE    S    W
## 1054  220 1382  422
```

# 3. 층화임의표집된 표본으로부터 모수 추정하기 (Computing Statistics from a Stratified Random Sample)

```
data(agstrat)
names(agstrat)  # list the variable names
```

```
##  [1] "county"   "state"    "acres92"  "acres87"  "acres82"  "farms92"
##  [7] "farms87"  "farms82"  "largef92" "largef87" "largef82" "smallf92"
## [13] "smallf87" "smallf82" "region"   "rn"       "strwt"
```

```
agstrat[1:6,1:8] # take a look at the first 6 obsns from columns 1 to 8
```

```
##     county state acres92 acres87 acres82 farms92 farms87 farms82
## 1 PIERCE C    NE  297326  332862  319619     725     857     865
## 2 JENNINGS    IN  124694  131481  139111     658     671     751
## 3 WAYNE CO    OH  246938  263457  268434    1582    1734    1866
## 4 VAN BURE    MI  206781  190251  197055    1164    1278    1464
## 5  OZAUKEE    WI   78772   85201   89331     448     483     527
## 6 CLEARWAT    MN  210897  229537  213105     583     699     693
```

```
nrow(agstrat)  # number of rows, 300
```

```
## [1] 300
```

```
unique(agstrat$region) # take a look at the four regions, NC, NE, S, W
```

```
## [1] "NC" "NE" "S"  "W"
```

```
table(agstrat$region)  # number of counties in each stratum
```

```
##
##  NC  NE   S   W
## 103  21 135  41
```

```
# check that the sum of the weights equals the population size
sum(agstrat$strwt)  # 3078
```

```
## [1] 3078
```

### Set up information for the survey design

```
# create a variable containing population stratum sizes, for use in fpc (optional)
# popsize_recode gives popsize for each stratum
popsize_recode <- c('NC' = 1054, 'NE' = 220, 'S' = 1382, 'W' = 422)
# next statement substitutes 1054 for each 'NC', 220 for 'NE', etc.
agstrat$popsize <- popsize_recode[agstrat$region]
table(agstrat$popsize)  #check the new variable
```

```
##
##  220  422 1054 1382
##   21   41  103  135
```

```
# input design information for agstrat
dstr <- svydesign(id = ~1, strata = ~region, weights = ~strwt, fpc = ~popsize, data = agstrat)
dstr
```

```
## Stratified Independent Sampling design
## svydesign(id = ~1, strata = ~region, weights = ~strwt, fpc = ~popsize,
##     data = agstrat)
```

### Calculate the statistics using the design object

```
# calculate mean, SE and confidence interval
smean<-svymean(~acres92, dstr)
smean
```

```
##           mean    SE
## acres92 295561 16380
```

```
confint(smean, level=.95, df=degf(dstr)) # note that df = n-H = 300-4
```

```
##          2.5 %   97.5 %
## acres92 263325 327796.5
```

```
# calculate total, SE and CI
stotal<-svytotal(~acres92, dstr)
stotal
```

```
##             total       SE
## acres92 909736035 50417248
```

```
degf(dstr) # Show the degrees of freedom for the design
```

```
## [1] 296
```

```
# calculate confidence intervals using the degrees of freedom
confint(stotal, level=.95,df= degf(dstr))
```

```
##             2.5 %     97.5 %
## acres92 810514350 1008957721
```

### Alternative design specifications

```
# Get same result if omit weights argument since weight = popsize/n_h
dstrfpc <- svydesign(id = ~1, strata = ~region, fpc = ~popsize, data = agstrat)
svymean(~acres92, dstrfpc)
```

```
##           mean    SE
## acres92 295561 16380
```

```
# If you include weights but not fpc, get SE without fpc factor
dstrwt <- svydesign(id = ~1, strata = ~region, weights = ~strwt, data = agstrat)
svymean(~acres92, dstrwt)
```

```
##           mean    SE
## acres92 295561 17241
```

### Calculating stratum means and variances

```
# calculate mean and se of acres92 by regions
svyby(~acres92, by=~region, dstr, svymean, keep.var = TRUE)
```

```
##    region   acres92       se
## NC     NC 300504.16 16107.59
## NE     NE  97629.81 18149.49
## S       S 211315.04 18925.35
## W       W 662295.51 93403.65
```

```
# calculate total and se of acres92 by regions
svyby(~acres92, ~region, dstr, svytotal, keep.var = TRUE)
```

```
##    region   acres92       se
## NC     NC 316731380 16977399
## NE     NE  21478558  3992889
## S       S 292037391 26154840
## W       W 279488706 39416342
```

```
# formula calculations, using tapply
# variables sampsize and popsize were calculated earlier in the chapter
# calculate mean within each region
strmean<-tapply(agstrat$acres92,agstrat$region,mean)
strmean
```

```
##        NC        NE         S         W
## 300504.16  97629.81 211315.04 662295.51
```

```
# calculate variance within each region
strvar<-tapply(agstrat$acres92,agstrat$region,var)
strvar
```

```
##           NC           NE            S            W
##  29618183543   7647472708  53587487856 396185950266
```

```
# verify standard errors by direct formula
strse<- sqrt((1-sampsize/popsize)*strvar/sampsize)
# same standard errors as from svyby
strse
```

```
##
##       NC       NE        S        W
## 16107.59 18149.49 18925.35 93403.65
```

## 4. 모비율의 추정 (Estimating proportions from a stratified random sample)

### Option 1: Use a 0-1 variable and find its mean

```
# Create variable lt200k
agstrat$lt200k <- rep(0,nrow(agstrat))
agstrat$lt200k[agstrat$acres92 < 200000] <- 1
# Rerun svydesign because the data set now has a new variable
dstr <- svydesign(id = ~1, strata = ~region, fpc = ~popsize,
                  weights = ~strwt, data = agstrat)
# calculate proportion, SE and confidence interval
smeanp<-svymean(~lt200k, dstr)
smeanp
```

```
##           mean     SE
## lt200k 0.51391 0.0248
```

```
confint(smeanp, level=.95,df=degf(dstr))
```

```
##            2.5 %    97.5 %
## lt200k 0.4651188 0.5627107
```

```
# calculate total, SE and CI
stotalp<-svytotal(~lt200k, dstr)
stotalp
```

```
##         total     SE
## lt200k 1581.8 76.318
```

```
confint(stotalp, level=.95,df=degf(dstr))
```

```
##           2.5 %   97.5 %
## lt200k 1431.636 1732.024
```

### Option 2: Create a factor variable lt200kf

```
agstrat$lt200kf <- factor(agstrat$lt200k)
# Rerun svydesign because the data set now has a new variable
dstr <- svydesign(id = ~1, strata = ~region, fpc = ~popsize,
                  weights = ~strwt, data = agstrat)
# calculate proportion, SE and confidence interval
smeanp2<-svymean(~lt200kf, dstr)
smeanp2
```

```
##             mean     SE
## lt200kf0 0.48609 0.0248
## lt200kf1 0.51391 0.0248
```

```
confint(smeanp2, level=.95,df=degf(dstr))
```

```
##              2.5 %    97.5 %
## lt200kf0 0.4372893 0.5348812
## lt200kf1 0.4651188 0.5627107
```

```
# calculate total, SE and CI
stotalp2<-svytotal(~lt200kf, dstr)
stotalp2
```

```
##           total     SE
## lt200kf0 1496.2 76.318
## lt200kf1 1581.8 76.318
```

```
confint(stotalp2, level=.95,df=degf(dstr))
```

```
##             2.5 %   97.5 %
## lt200kf0 1345.976 1646.364
## lt200kf1 1431.636 1732.024
```

### Option 3: Construct asymmetric confidence intervals

```
# calculate proportion and confidence interval with svyciprop
svyciprop(~I(lt200k==1), dstr, method="beta")
```

```
##                       2.5% 97.5%
## I(lt200k == 1) 0.514 0.464  0.56
```