Sampling and Estimation - Exercise 2

Stefan Zins, Matthias Sand and Jan-Philipp Kolb

2 February 2016

Exercise 2.A

Estimation under a stratified design

- Download the ESS for Sweden and Denmark (round 5)
- Import data to R and combine the two datasets
- Define a `survey` object (stratified design)
- Estimate the empirical distribution of tv consumption (`tvtot`) in Sweden and Denmark
- Estimate the joined empirical distribution of tv consumption in Sweden and Denmark

Solution Exercise 2.A

- Download the ESS for Sweden and Denmark (round 5)

Use the package `foreign` for import

```r
library(foreign)
```

Load the ESS dataset and the country file

```r
DK <- read.spss("ESS5DK.sav",to.data.frame=T)
SE <- read.spss("ESS5SE.sav",to.data.frame=T)
```

Construction of N

Later it is necessary for the specification of fpc.

```r
DK$N <- DK$dweight*DK$pweight*10000*nrow(DK)

SE$N <- SE$dweight*SE$pweight*10000*nrow(SE)
```

Combine the two datasets (DK,SE)

```r
DK_tv <- data.frame(tvtot=as.character(DK$tvtot),
                    N=DK$N,
                    cntry=as.character(DK$cntry))
```

```r
SE_tv <- data.frame(tvtot=as.character(SE$tvtot),
                    N=SE$N,
                    cntry=as.character(SE$cntry))
```

```r
NE <- rbind(DK_tv,SE_tv)
```

Adding new variables

```r
NE$mt3 <- 0
NE$mt3[NE$tvtot=="More than 3 hours"] <- 1
```

R-package `survey`

```r
library(survey)
```

- Define survey objects

```r
# SRS design
svydes_DK <- svydesign(id=~1,fpc=~N, data=DK)
svydes_SE <- svydesign(id=~1,fpc=~N, data=SE)
```

```r
# Stratified design
svydes_NE <- svydesign(id=~1,strata=~cntry,
                       fpc=~N, data=NE)
```

The analysis functions in R-package `survey`

- Estimate the empirical distribution of tv consumption (`tvtot`) in Sweden and Denmark

```r
stab_DK <- svytable(~tvtot,svydes_DK)
stab_DK
```

```r
stab_SE <- svytable(~tvtot,svydes_SE)
stab_SE
```

empirical distribution of joined dataset

```r
stab_NE <- svytable(~tvtot,svydes_NE)
stab_NE
```

Visualisation

```r
# R-package for visualisation

library(lattice)
barchart(stab_DK)
barchart(stab_SE)
```

More analysis functions

Estimate the number of persons watching 3 or more hours:

```
svytotal(~mt3,svydes_NE)
```

Percentage of persons watching more than 3 hours tv:

```
svymean(~mt3,svydes_NE)
```

Exercise 2.B

- Load the `survey` package and the `api` datasets.

```
library(survey)
data(api)
```

The dataset apipop

- The dataset `apistrat` is a sample of schools from `apipop` stratified by `stype`.

cds	stype	name	sname	snum
01611190130229	H	Alameda High	Alameda High	1
01611190132878	H	Encinal High	Encinal High	2
01611196000004	M	Chipman Middle	Chipman Middle	3
01611196090005	E	Lum (Donald D.)	Lum (Donald D.) Elementary	4
01611196090013	E	Edison Elementa	Edison Elementary	5
01611196090021	E	Otis (Frank) El	Otis (Frank) Elementary	6
01611196090039	E	Franklin Elemen	Franklin Elementary	7
01611196090047	E	Haight Elementa	Haight Elementary	8

Computing the mean with survey package

- Assuming the selection within the strata was done by SRS, define a survey object (`svydesign`) and calculate a point and variance estimate for the mean of `api00`.

```
svy_api <- svydesign(id=~1,strata=~stype,
                     fpc=~fpc, data=apistrat)

svymean(~api00,svy_api)
```

```
##         mean     SE
## api00 662.29 9.4089
```

Exercise 2.B

- Using `stype` again as a stratification variable try different allocations for stratified sample. Calculate the allocation of a sample of 60 schools from `apipop` using equal, proportional and optimal allocation. The proportional allocation should be proportional to the number of schools within the strata and the optimal alloaction should be optimal with regard to `api99`.

Function for stratified samples

```
strSRsample <- function(strind, nh, replace=FALSE){
  Nh <- table(strind)[names(nh)]
  h.id <- split(1:sum(Nh), strind)[names(nh)]


  sam <- mapply( function(x,y) sample(x, y,
                 replace=replace)
                 , Nh, nh, SIMPLIFY = F)
  unlist(mapply(function(x,y) x[y]
                , h.id
                , sam, SIMPLIFY = F)
         ,use.names = FALSE)
}
```

Getting the function

```
library(devtools)
install_github("BernStZi/SamplingAndEstimation/r/
               sampaest",
               ref="short")
```

```
url <- "http://raw.githubusercontent.com/BernStZi/
SamplingAndEstimation/short/r/sampaest/R/strSRsample.R"
source(url)
```

Equal allocation:

```
nh.eq <- c(20,20,20)
names(nh.eq) <- names(table(apipop$stype))

s.eq <- strSRsample(apipop$stype, nh.eq, replace=FALSE)
```

Proportional allocation:

```
Nh.tab  <- table(apipop$stype)
n <- 60
nh.pr <- round(Nh.tab/sum(Nh.tab)*n)

s.pr <- strSRsample(apipop$stype, nh.pr, replace=FALSE)
```

Optimal allocation:

```
V.h   <- tapply(apipop$api99,apipop$stype,sd)[names(Nh.tab)]
nh.op <- round((Nh.tab*V.h)/(sum(Nh.tab*V.h))*n)
```

```
s.op <- strSRsample(apipop$stype, nh.op, replace=FALSE)
```

Exercise 2.B

- Select a StrSRS from `apipop` for each allocation.

Subselect the sample:

```
pop <- apipop
pop$Nh <- table(apipop$stype)[apipop$stype]
strSRS.eq <- pop[s.eq,]
strSRS.pr <- pop[s.pr,]
strSRS.op <- pop[s.op,]
```

Equal allocation

- Estimate again the mean of `api00` from all three samples and compare the results.

```
svystrSRS.eq <- svydesign(ids=~cds, strata =~stype,
                          fpc=~Nh, data=strSRS.eq)

svymean(~api00, svystrSRS.eq)
```

```
##          mean     SE
## api00 634.47 23.235
```

Proportional allocation

```
svystrSRS.pr <- svydesign(ids=~cds, strata =~stype,
                          fpc=~Nh, data=strSRS.pr)

svymean(~api00, svystrSRS.pr)
```

```
##          mean     SE
## api00 673.13 16.497
```

Optimal allocation

```r
svystrSRS.op <- svydesign(ids=~cds, strata =~stype,
                          fpc=~Nh, data=strSRS.op)

svymean(~api00, svystrSRS.op)
```

```
##         mean     SE
## api00 667.68 15.766
```