Exploratory Data Analysis with R

Sampling Techniques and Estimation

Xuemao Zhang
East Stroudsburg University

October 17, 2022

Outline

- Sampling and estimation
  - Simple random sampling
  - Systematic sampling
  - Stratified sampling
  - Clustered sampling

Sampling

- **Population**: The complete collection of subjects we want to study. It is a finite set of labeled individuals.

$$U = \{1, 2, \ldots, N\}$$

- Population **parameters** like population mean, variance and proportion are unknown unless a census is conducted.
- **Sample**: A subset of the population with study variable(s) measured on each selected individuals.

$$s = \{1, 2, \ldots, n\}$$

  - Sampling unit: The unit we actually sample.
- **Sampling frame**: A list, map, or other specification of sampling units in the population from which a sample may be selected.
- **Probability Sampling**: each element (sampling unit) in the (study) population has a known, non-zero probability of being included in the sample.

Simple random sampling without replacement

- SRS without replacement: select the 1st element from $U = \{1, 2, \ldots, N\}$ with probability $1/N$; select the 2nd element from the remaining $N - 1$ elements with probability $1/(N - 1)$; and continue this until n elements are selected.
    - Every possible subset of n distinct units in the population has the same probability of being selected as the sample.
- The R base function `sample()` can take a sample using either with or without replacement:
    - **usage**

```
sample(x, size, replace = FALSE, prob = NULL)
```

- Package `sampling` implements the selection of many sampling designs.
    - We use the `sampling` package to conduct sampling.
- Package `survey` allows to specify a complex survey design and conduct statistical estimations.
    - We use the `survey` package to conduct estimations.
- Statistical analysis: We will focus on the estimation of population mean or proportion based on probability samples.

SRS without replacement

- Consider the data `rec99` in the package `sampling` which contains an extract from the last French Census performed in 1999.
- Data is collected on $N = 554$ French towns from the south of France (the region Haute-Garonne) about:
  - `POPSDC99`: the number of habitants in 1999
  - `LOG`: the number of dwellings
  - `LOGVAC`: the number of empty dwellings

```
library(sampling)
data("rec99", package="sampling")
dim(rec99)
```

```
## [1] 554  10
```

```
str(rec99)
```

```
## 'data.frame':    554 obs. of  10 variables:
##  $ CODE_N    : int  31001 31002 31003 31004 31005 31006 31007 31008 3100
##  $ COMMUNE   : chr  "AGASSAC" "AIGNES" "AIGREFEUILLE" "AYGUESVIVES" ...
##  $ BVQ_N     : int  31239 31033 31044 31582 31028 31106 31239 31239 3148
##  $ POPSDC99  : int  121 193 577 1815 299 159 72 230 84 100 ...
##  $ LOG       : int  65 99 179 656 163 60 37 120 77 76 ...
##  $ LOGVAC    : int  9 7 1 15 9 3 4 5 3 0 ...
```

SRS without replacement

- selection of a simple random sampling without replacement of size 70
  - Using the `sample` function

```
n=70
N=554
subindex=sample(1:N, size=n, replace = FALSE)
#sample(1:N, size=n, replace = FALSE, prob = rep(n/N,N))
data0=rec99[subindex,]
```

- Using the `srswor` function in package `sampling`

```
si.rec<-srswor(n,N) #1 means included; 0 means excluded
data1=rec99[which(si.rec==1),]; # selected towns
#which() returns the row numbers
```

- For SRS without replacement
  - fpc (finite population correction) $= n/N$
  - sampling weight $= N/n$ for each selected unit

SRS without replacement

- Function `survey::svydesign`
  - id : Formula or data frame specifying cluster ids from largest level to smallest level, ~0 or ~1 is a formula for no clusters.
  - strata: stratification variable;
  - weights : formula or vector of inclusion probabilities of size equal to sample size;
  - fpc : formula or vector with the finite population correction (same size as weights); if fpc not specified, then the sample is with replacement;
  - data : the sample data.

```
library(survey)
data1$fpc=n/N
data1$wt=N/n
data1_SRS=svydesign(data=data1,ids=~1,fpc=~fpc,weights=~wt)
```

SRS without replacement

- data1_SRS is an object and information

```
attributes(data1_SRS)
```

```
## $names
## [1] "cluster"    "strata"     "has.strata" "prob"       "allprob'
## [6] "call"       "variables"  "fpc"        "pps"
##
## $class
## [1] "survey.design2" "survey.design"
```

```
summary(data1_SRS)
```

```
## Independent Sampling design
## svydesign(data = data1, ids = ~1, fpc = ~fpc, weights = ~wt)
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1264  0.1264  0.1264  0.1264  0.1264  0.1264
## Population size (PSUs): 554
## Data variables:
##  [1] "CODE_N"      "COMMUNE"     "BVQ_N"       "POPSDC99"    "LOG"
```

SRS without replacement

- Estimation of the total and mean and standard-deviation for the SRS sample for the variable `LOGVAC`

```
svy_total=svytotal(~LOGVAC, data1_SRS)
svy_total
```

```
##         total     SE
## LOGVAC 15646 3366.6
```

```
svy_mean=svymean(~LOGVAC, data1_SRS)
svy_mean
```

```
##           mean     SE
## LOGVAC 28.243 6.0769
```

- Confidence interval estimation

```
confint(svy_total, level=0.95)
```

```
##            2.5 %   97.5 %
## LOGVAC 9048.115 22244.97
```

```
confint(svy_mean, level=0.95)
```

```
##            2.5 %   97.5 %
```

SRS without replacement

- If there is a binary variable in the data set, we can estimate the proportion parameter.
- To produce confidence intervals more accurate for proportions, we use the function `survey::svyciprop`.

SRS with replacement

- SRS with replacement: Select the 1st element from $U = \{1, 2, \ldots, N\}$ with equal probability; select the 2nd element also from $U = \{1, 2, \ldots, N\}$ with equal probability; repeat this n times.
- Under SRS with replacement, some units in the population may be selected more than once.
- SRS without replacement is more efficient than SRS with replacement.
- When N is very large and n is small, the two sampling methods will be very close to each other.

SRS with replacement

- Using the sample function

```
n=70
N=554
subindex=sample(1:N, n, replace=TRUE)
subindex
```

```
##  [1] 443  39  85 123 163  90 342 197 490 470 316 189  50 523   4 263 132
## [20] 212 143 113 525 295 357 265  27 176 404 266 230  67 296 193  42 225
## [39] 490 177 358 113 480 446 419 158  13 210 430 326 521 289 519 488  31
## [58] 391 122 291  69 355 417  40 168 522 314 425  87  30
```

```
data0=rec99[subindex,]
str(data0)
```

```
## 'data.frame':    70 obs. of  10 variables:
##  $ CODE_N    : int  31475 31042 31091 31135 31178 31096 31369 31215 3152
##  $ COMMUNE   : chr  "SAINT-CLAR-D" "BAGNERES-DE-" "BRUGUIERES" "CAZERES"
##  $ BVQ_N     : int  31395 31042 31555 31135 31239 31098 31483 31573 3110
##  $ POPSDC99  : int  872 2900 3862 3260 214 128 36 495 301 235 ...
##  $ LOG       : int  352 4490 1395 1761 119 66 29 188 135 98 ...
##  $ LOGVAC    : int  16 350 32 214 14 10 1 7 5 2 ...
##  $ STRATLOG  : int  3 4 4 4 2 1 1 2 2 1 ...
```

SRS with replacement

- Using packages sampling and survey

```
si.wr<-srswr(n,N) # 0 means excluded
data2=rec99[which(si.wr!=0),] # selected towns
dim(data2)  #sample size is decreased
```

```
## [1] 67 10
```

```
# To specify sampling with replacement, simply omit the fpc argumen
data2_SRS=svydesign(data=data2,ids=~1,
                    weights=~rep(N/dim(data2)[1], dim(data2)[1]))
summary(data2_SRS)
```

```
## Independent Sampling design (with replacement)
## svydesign(data = data2, ids = ~1, weights = ~rep(N/dim(data2)[1],
##     dim(data2)[1]))
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1209  0.1209  0.1209  0.1209  0.1209  0.1209
## Data variables:
##  [1] "CODE_N"     "COMMUNE"    "BVQ_N"      "POPSDC99"   "LOG"
##  [6] "LOGVAC"     "STRATLOG"   "surf_m2"    "lat_centre" "lon_cen
```

SRS with replacement

- Estimation of total and mean

```
svy_total=svytotal(~LOGVAC, data2_SRS)
svy_total
```

```
##          total     SE
## LOGVAC 8657.3 2383.9
```

```
svy_mean=svymean(~LOGVAC, data2_SRS)
svy_mean
```

```
##          mean     SE
## LOGVAC 15.627 4.3031
```

```
confint(svy_total, level=0.95)
```

```
##            2.5 %   97.5 %
## LOGVAC 3984.921 13329.65
```

```
confint(svy_mean, level=0.95)
```

```
##            2.5 %   97.5 %
## LOGVAC 7.192998 24.06073
```

Systematic Sampling

- When no list of population exists or when the list is in roughly random order.
  - Systematic sampling is used as a proxy for simple random sampling.
- To choose a sample of size n: Let $k = N/n$ if an integer or let k be the next integer after N/n.
  - Choose a random starting point r, $1 \leq r \leq k$;
  - The elements numbered r, $r + k$, $r + 2k$, $\ldots$, $r + (n-1)k$ will form the sample.
- Function `UPsystematic(pik)` implements the systematic sampling with unequal inclusion probabilities contained in the N-dimensional vector `pik`.

```
pik=rep(n/N,N)
sys.rec=UPsystematic(pik)
data3=rec99[which(sys.rec==1),]
```

Systematic Sampling

- Estimation using the method with SRS without replacement

```
data3$fpc=n/N
data3$wt=N/n
data3_Sys=svydesign(data=data3,ids=~1,fpc=~fpc,weights=~wt)
summary(data3_Sys)
```

```
## Independent Sampling design
## svydesign(data = data3, ids = ~1, fpc = ~fpc, weights = ~wt)
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1264  0.1264  0.1264  0.1264  0.1264  0.1264
## Population size (PSUs): 554
## Data variables:
##  [1] "CODE_N"      "COMMUNE"     "BVQ_N"       "POPSDC99"    "LOG"
##  [6] "LOGVAC"      "STRATLOG"    "surf_m2"     "lat_centre"  "lon_cen
## [11] "fpc"         "wt"
```

Systematic Sampling

- Estimation using the method with SRS without replacement

```
svy_total=svytotal(~LOGVAC, data3_Sys)
svy_total
```

```
##          total     SE
## LOGVAC 6798.4 980.05
```

```
svy_mean=svymean(~LOGVAC, data3_Sys)
svy_mean
```

```
##           mean     SE
## LOGVAC 12.271 1.7691
```

```
confint(svy_total, level=0.95)
```

```
##          2.5 %   97.5 %
## LOGVAC 4877.5 8719.243
```

```
confint(svy_mean, level=0.95)
```

```
##            2.5 %   97.5 %
## LOGVAC 8.804152 15.73871
```

Stratified Sampling

- Sometimes the population is naturally divided into a number of distinct non-overlapping subpopulations called **strata**.
- Stratified Sampling method: taking one simple random sample from each stratum independently.
  - We must know the strata information before sampling.
- We may want data of known precision for subgroups of the population. These subgroups should be the strata.
- A stratified sample may be more convenient to administer and may result in a lower cost for the survey.
- Stratified sampling often gives more precise (having lower variance) estimates for population means and totals.

Stratified Sampling

- A stratified sample may be selected by using the function `sampling::strata`
- The population is divided into four strata by the variable `stratlog`.
  - To select a stratified sample, we need to give the sample sizes to select within each stratum.

```
table(rec99$STRATLOG)
```

```
##
##   1   2   3   4
## 221 169 110  54
```

```
sizes=c(28,21,14,7) #proportional to the strata sizes
s=sampling::strata(data=rec99,stratanames="STRATLOG",
                  size=sizes,method="srswor")
# extracts the observed data
data4=getdata(rec99,s)
str(data4)
```

```
## 'data.frame':    70 obs. of  13 variables:
##  $ CODE_N    : int  31009 31010 31012 31014 31018 31055 31064 311
##  $ COMMUNE   : chr  "ANTICHAN-DE-" "ANTIGNAC" "ARBON" "ARGUENOS"
```

Stratified Sampling

- Estimation
  - fpc in stratum i $= n_i/N_i$
  - weight in stratum i $= N_i/n_i$

```
N1=221;N2=169;N3=110;N4=54;
n1=sizes[1];n2=sizes[2];n3=sizes[3];n4=sizes[4];
wt_str=c(rep(N1/n1,n1),rep(N2/n2,n2),rep(N3/n3,n3),rep(N4/n4,n4))
fpc_str=c(rep(n1/N1,n1),rep(n2/N2,n2),rep(n3/N3,n3),rep(n4/N4,n4))
data4_str=svydesign(data=data4,ids=~1,strata=~STRATLOG,
          fpc=~fpc_str, weights=~wt_str)
summary(data4_str)
```

```
## Stratified Independent Sampling design
## svydesign(data = data4, ids = ~1, strata = ~STRATLOG, fpc = ~fpc_
##     weights = ~wt_str)
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1243  0.1243  0.1267  0.1264  0.1273  0.1296
## Stratum Sizes:
##              1  2  3 4
## obs         28 21 14 7
```

Stratified Sampling

- Estimation
  - fpc in stratum i = n_i/N_i
  - weight in stratum i = N_i/n_i

```
svy_total=svytotal(~LOGVAC, data4_str); svy_total;
```

```
##          total     SE
## LOGVAC 10879 1354.3
```

```
svy_mean=svymean(~LOGVAC, data4_str); svy_mean;
```

```
##          mean     SE
## LOGVAC 19.637 2.4446
```

```
confint(svy_total, level=0.95)
```

```
##           2.5 %   97.5 %
## LOGVAC 8224.531 13533.35
```

```
confint(svy_mean, level=0.95)
```

```
##           2.5 %   97.5 %
## LOGVAC 14.84572 24.42843
```

Cluster Sampling

- In many practical situations the population elements are grouped into a number of clusters.
  - The population may be widely distributed geographically or may occur in natural clusters such as households or schools, and it is less expensive to take a sample of clusters rather than an SRS of individuals
- Cluster sampling: the total population is divided into clusters and a simple random sample of these clusters is selected. The elements in each cluster are then sampled.
  - One-stage cluster sampling: Take n clusters using simple random sampling without replacement, and all elements in the selected clusters are observed.
  - Two-stage cluster sampling: after selecting n clusters, then selecting a probability sample of elements from each cluster.

Cluster Sampling

- A cluster sample may be selected by using the function `sampling::cluster` and `sampling::mstage`
- The data `rec99` has 32 clusters by the varialble `BVQ_N` (bassin de vie quotidien)

```
unique(rec99$BVQ_N)
```

```
##  [1] 31239 31033 31044 31582 31028 31106 31483 31042 31523 31020
## [13] 31395 31390 31232 31113 31144 31080 31375 31358 31454 31561
## [25] 31098 31107 31584 31573 31499 31202 31135 31193
```

- Let's consider one-stage cluster sampling

Cluster Sampling

- We select $n_c = 8$ clusters by simple random sampling without replacement.

```
cl=sampling::cluster(rec99,clustername="BVQ_N",size=8,
                     method="srswor")
#getting the cluster sample data
data5=getdata(rec99,cl)
dim(data5)
```

```
## [1] 118  12
```

```
#str(data5)
```

```
data5_cl=svydesign(data=data5,ids=~BVQ_N,
          weights=~rep(32/8,nrow(data5)),
          fpc=~rep(8/32,nrow(data5)))
summary(data5_cl)
```

```
## 1 - level Cluster Sampling design
## With (8) clusters.
## svydesign(data = data5, ids = ~BVQ_N, weights = ~rep(32/8, nrow(d
##     fpc = ~rep(8/32, nrow(data5)))
## Probabilities:
```

Cluster Sampling

- Estimation

```
svy_total=svytotal(~LOGVAC, data5_cl); svy_total;
```

```
##         total   SE
## LOGVAC  7648 1731
```

```
svy_mean=svymean(~LOGVAC, data5_cl); svy_mean;
```

```
##           mean     SE
## LOGVAC 16.203 2.0319
```

```
confint(svy_total, level=0.95)
```

```
##            2.5 %   97.5 %
## LOGVAC 4255.338 11040.66
```

```
confint(svy_mean, level=0.95)
```

```
##            2.5 %  97.5 %
## LOGVAC 12.22098 20.1858
```

License