ch06output

2024-04-10

```
##### Install the R packages needed for the chapter

library(survey)
```

```
## Loading required package: grid

## Loading required package: Matrix

## Loading required package: survival

##
## Attaching package: 'survey'

## The following object is masked from 'package:graphics':
##
##     dotchart
```

```
library(sampling)
```

```
##
## Attaching package: 'sampling'

## The following objects are masked from 'package:survival':
##
##     cluster, strata
```

```
library(SDAResources)

########## Selecting a Sample with Unequal Probabilities ##########

##### Example 6.2

data(classes)
classes[1:6,]
```

```
##   class class_size
## 1     1         44
## 2     2         33
## 3     3         26
## 4     4         22
## 5     5         76
## 6     6         63
```

```r
N<-nrow(classes)
set.seed(78065)
# select 5 classes with probability proportional to class size and with replacement
sample_units<-sample(1:N,5,replace=TRUE,prob=classes$class_size)
sample_units
```

```
## [1]  5 14  6 14  6
```

```r
mysample<-classes[sample_units,]
mysample
```

```
##      class class_size
## 5        5         76
## 14      14        100
## 6        6         63
## 14.1    14        100
## 6.1      6         63
```

```r
# calculate ExpectedHits and sampling weights
mysample$ExpectedHits<-5*mysample$class_size/sum(classes$class_size)
mysample$SamplingWeight<-1/mysample$ExpectedHits
mysample$psuid<-row.names(mysample)
mysample
```

```
##      class class_size ExpectedHits SamplingWeight psuid
## 5        5         76    0.5873261       1.702632     5
## 14      14        100    0.7727975       1.294000    14
## 6        6         63    0.4868624       2.053968     6
## 14.1    14        100    0.7727975       1.294000  14.1
## 6.1      6         63    0.4868624       2.053968   6.1
```

```r
# check sum of sampling weights
sum(mysample$SamplingWeight)
```

```
## [1] 8.398568
```

```r
# sampling without replacement
set.seed(330582)
cluster(data=classes, clustername=c("class"), size=5, method="systematic",
        pik=classes$class_size,description=TRUE)
```

```
## Number of selected clusters: 5
## Number of units in the population and number of selected units: 15 5
```

```
##   class ID_unit      Prob
## 1     1       1 0.3400309
## 2     5       5 0.5873261
## 3     8       8 0.3400309
## 4    11      11 0.3554869
## 5    14      14 0.7727975
```

```
########## Selecting a Two-Stage Cluster Sample ##########

##### Example 6.11

# create data frame classeslong
data(classes)
classeslong<-classes[rep(1:nrow(classes),times=classes$class_size),]
classeslong$studentid <- sequence(classes$class_size)
nrow(classeslong)
```

```
## [1] 647
```

```
table(classeslong$class) # check class sizes
```

```
##
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##  44  33  26  22  76  63  20  44  54  34  46  24  46 100  15
```

```
head(classeslong)
```

```
##     class class_size studentid
## 1       1         44         1
## 1.1     1         44         2
## 1.2     1         44         3
## 1.3     1         44         4
## 1.4     1         44         5
## 1.5     1         44         6
```

```
# select a two-stage cluster sample, psu: class, ssu: studentid
# number of psus selected: n = 5 (pps systematic)
# number of students selected: m_i = 4 (srs without replacement)
# problist<-list(classes$class_size/647) # same results as next command
problist<-list(classes$class_size/647,4/classeslong$class_size) #selection prob
problist[[1]]  # extract the first object in the list. This is pps, size M_i/M
```

```
##  [1] 0.06800618 0.05100464 0.04018547 0.03400309 0.11746522 0.09737249
##  [7] 0.03091190 0.06800618 0.08346213 0.05255023 0.07109737 0.03709428
## [13] 0.07109737 0.15455951 0.02318393
```

```
problist[[2]][1:5] # first 5 values in second object in list
```

```
## [1] 0.09090909 0.09090909 0.09090909 0.09090909 0.09090909
```

```
# number of psus and ssus
n<-5
numberselect<-list(n,rep(4,n))
numberselect
```

```
## [[1]]
## [1] 5
##
## [[2]]
## [1] 4 4 4 4 4
```

```
# two-stage sampling
set.seed(75745)
tempid<-mstage(classeslong,stage=list("cluster","stratified"),
               varnames=list("class","studentid"),
               size=numberselect, method=list("systematic","srswor"),pik=problist)

# get data
sample1<-getdata(classeslong,tempid)[[1]]
# sample 1 contains the ssus of the 5 psus chosen at the first stage
# Prob_ 1 _stage has the first-stage selection probabilities
head(sample1)
```

```
##      class_size studentid class ID_unit Prob_ 1 _stage
## 4.21         22        22     4     125      0.1700155
## 4.20         22        21     4     124      0.1700155
## 4.6          22         7     4     110      0.1700155
## 4            22         1     4     104      0.1700155
## 4.7          22         8     4     111      0.1700155
## 4.8          22         9     4     112      0.1700155
```

```
nrow(sample1)
```

```
## [1] 285
```

```
table(sample1$class) # lists the psus selected in the first stage
```

```
##
##   4   6   9  13  14
##  22  63  54  46 100
```

```
sample2<-getdata(classeslong,tempid)[[2]]
# sample 2 contains the final sample
# Prob_ 2 _stage has the second-stage selection probabilities
# Prob has the final selection probabilities
head(sample2)
```

```
##      class class_size studentid ID_unit Prob_ 2 _stage      Prob
## 4.21     4         22        22     125     0.18181818 0.0309119
## 4.7      4         22         8     111     0.18181818 0.0309119
## 4.5      4         22         6     109     0.18181818 0.0309119
## 4.19     4         22        20     123     0.18181818 0.0309119
## 6.48     6         63        49     250     0.06349206 0.0309119
## 6.53     6         63        54     255     0.06349206 0.0309119
```

```r
nrow(sample2) # sample of 20 ssus altogether
```

```
## [1] 20
```

```r
table(sample2$class) # 4 ssus selected from each psu
```

```
##
##  4  6  9 13 14
##  4  4  4  4  4
```

```r
# calculate final weight = 1/Prob
sample2$finalweight<-1/sample2$Prob
# check that sum of final sampling weights equals population size
sum(sample2$finalweight)
```

```
## [1] 647
```

```r
sample2[,c(1,2,3,6,7)] # print variables from final sample
```

```
##       class class_size studentid      Prob finalweight
## 4.21      4         22        22 0.0309119       32.35
## 4.7       4         22         8 0.0309119       32.35
## 4.5       4         22         6 0.0309119       32.35
## 4.19      4         22        20 0.0309119       32.35
## 6.48      6         63        49 0.0309119       32.35
## 6.53      6         63        54 0.0309119       32.35
## 6.23      6         63        24 0.0309119       32.35
## 6.33      6         63        34 0.0309119       32.35
## 9.50      9         54        51 0.0309119       32.35
## 9.29      9         54        30 0.0309119       32.35
## 9.31      9         54        32 0.0309119       32.35
## 9.36      9         54        37 0.0309119       32.35
## 13.10    13         46        11 0.0309119       32.35
## 13       13         46         1 0.0309119       32.35
## 13.45    13         46        46 0.0309119       32.35
## 13.39    13         46        40 0.0309119       32.35
## 14.4     14        100         5 0.0309119       32.35
## 14.78    14        100        79 0.0309119       32.35
## 14.98    14        100        99 0.0309119       32.35
## 14.63    14        100        64 0.0309119       32.35
```

```r
### use Sampford's method

# select a cluster sample in two stages, psu: class, ssu: studentid
# number of psu selected n =5 (Sampford's method)
# first, convert the measure of size to a vector of probabilities
classes$stage1prob<-inclusionprobabilities(classes$class_size,5)
sum(classes$stage1prob)  # inclusion probabilities sum to n
```

```
## [1] 5
```

```
# select the psus
set.seed(29385739)
stage1.units<-UPsampford(classes$stage1prob)
stage1.sample<-getdata(classes,stage1.units)
stage1.sample
```

```
##    ID_unit class class_size stage1prob
## 1        1     1         44  0.3400309
## 3        3     3         26  0.2009274
## 7        7     7         20  0.1545595
## 13      13    13         46  0.3554869
## 14      14    14        100  0.7727975
```

```
# first-stage units are in stage1.sample
# now select the second-stage units (students)
# convert the psus in the sample to long format and assign student ids
npsu<-nrow(stage1.sample)
stage1.long<-stage1.sample[rep(1:npsu,times=stage1.sample$class_size),]
stage1.long$studentid<-sequence(stage1.sample$class_size)
head(stage1.long)
```

```
##     ID_unit class class_size stage1prob studentid
## 1         1     1         44  0.3400309         1
## 1.1       1     1         44  0.3400309         2
## 1.2       1     1         44  0.3400309         3
## 1.3       1     1         44  0.3400309         4
## 1.4       1     1         44  0.3400309         5
## 1.5       1     1         44  0.3400309         6
```

```
# use strata function to select 4 ssus from each psu
stage2.units<-strata(stage1.long, stratanames=c("class"),
                     size=rep(4,5), method="srswor")
nrow(stage2.units)
```

```
## [1] 20
```

```
# get the data for the second-stage sample
ssusample<-getdata(stage1.long,stage2.units)
head(ssusample)
```

```
##      class_size stage1prob studentid class ID_unit       Prob Stratum
## 1.3          44  0.3400309         4     1       4 0.09090909       1
## 1.13         44  0.3400309        14     1      14 0.09090909       1
## 1.21         44  0.3400309        22     1      22 0.09090909       1
## 1.26         44  0.3400309        27     1      27 0.09090909       1
## 3.11         26  0.2009274        12     3      56 0.15384615       2
## 3.18         26  0.2009274        19     3      63 0.15384615       2
```

```
# compute the sampling weights
# stage1prob contains stage 1 sampling probability;
# Prob has stage 2 sampling probability
```

```
ssusample$finalprob<- ssusample$stage1prob*ssusample$Prob
ssusample$finalwt<-1/ssusample$finalprob
sum(ssusample$finalwt)  # check sum of weights
```

```
## [1] 647
```

```
# print selected columns of ssusample
print(ssusample[,c(1,2,3,4,6,8,9)],digits=4)
```

```
##       class_size stage1prob studentid class    Prob finalprob finalwt
## 1.3           44     0.3400         4     1 0.09091   0.03091   32.35
## 1.13          44     0.3400        14     1 0.09091   0.03091   32.35
## 1.21          44     0.3400        22     1 0.09091   0.03091   32.35
## 1.26          44     0.3400        27     1 0.09091   0.03091   32.35
## 3.11          26     0.2009        12     3 0.15385   0.03091   32.35
## 3.18          26     0.2009        19     3 0.15385   0.03091   32.35
## 3.19          26     0.2009        20     3 0.15385   0.03091   32.35
## 3.24          26     0.2009        25     3 0.15385   0.03091   32.35
## 7.11          20     0.1546        12     7 0.20000   0.03091   32.35
## 7.13          20     0.1546        14     7 0.20000   0.03091   32.35
## 7.18          20     0.1546        19     7 0.20000   0.03091   32.35
## 7.19          20     0.1546        20     7 0.20000   0.03091   32.35
## 13.16         46     0.3555        17    13 0.08696   0.03091   32.35
## 13.31         46     0.3555        32    13 0.08696   0.03091   32.35
## 13.34         46     0.3555        35    13 0.08696   0.03091   32.35
## 13.42         46     0.3555        43    13 0.08696   0.03091   32.35
## 14.1         100     0.7728         2    14 0.04000   0.03091   32.35
## 14.20        100     0.7728        21    14 0.04000   0.03091   32.35
## 14.35        100     0.7728        36    14 0.04000   0.03091   32.35
## 14.68        100     0.7728        69    14 0.04000   0.03091   32.35
```

```
########## Computing Estimates from an Unequal-Probability Sample ##########


##### Example 6.4

studystat <- data.frame(class = c(12, 141, 142, 5, 1),
                        Mi = c(24, 100, 100, 76, 44),
                        tothours=c(75,203,203,191,168))
studystat
```

```
##   class  Mi tothours
## 1    12  24       75
## 2   141 100      203
## 3   142 100      203
## 4     5  76      191
## 5     1  44      168
```

```
studystat$wt<-647/(studystat$Mi*5)
sum(studystat$wt) # check weight sum, which estimates N=15 psus
```

```
## [1] 12.62321
```

```r
# design for with-replacement sample, no fpc argument
d0604 <- svydesign(id = ~1, weights=~wt, data = studystat)
d0604
```

```
## Independent Sampling design (with replacement)
## svydesign(id = ~1, weights = ~wt, data = studystat)
```

```r
# Ratio estimation using Mi as auxiliary variable
ratio0604<-svyratio(~tothours, ~Mi,design = d0604)
ratio0604
```

```
## Ratio estimator: svyratio.survey.design2(~tothours, ~Mi, design = d0604)
## Ratios=
##                Mi
## tothours 2.703268
## SEs=
##                 Mi
## tothours 0.3437741
```

```r
confint(ratio0604, level=.95,df=4)
```

```
##                2.5 %   97.5 %
## tothours/Mi 1.748798 3.657738
```

```r
# Can also estimate total hours studied for all students in population
svytotal(~tothours,d0604)
```

```
##          total     SE
## tothours  1749 222.42
```

```r
##### Example 6.6

students <- data.frame(class = rep(studystat$class,each=5),
   popMi = rep(studystat$Mi,each=5),
   sampmi=rep(5,25),
   hours=c(2,3,2.5,3,1.5,2.5,2,3,0,0.5,3,0.5,1.5,2,3,1,2.5,3,5,2.5,4,4.5,3,2,5))
# The 'with' function allows us to calculate using variables from a data frame
# without having to type the data frame name for all of them
students$studentwt <- with(students,(647/(popMi*5)) * (popMi/sampmi))
# check the sum of the weights
sum(students$studentwt)
```

```
## [1] 647
```

```r
# create the design object
d0606 <- svydesign(id = ~class, weights=~studentwt, data = students)
d0606
```

```
## 1 - level Cluster Sampling design (with replacement)
## With (5) clusters.
## svydesign(id = ~class, weights = ~studentwt, data = students)
```

```r
# estimate mean and SE
svymean(~hours,d0606)
```

```
##       mean     SE
## hours  2.5 0.3606
```

```r
degf(d0606)
```

```
## [1] 4
```

```r
confint(svymean(~hours,d0606),level=.95,df=4) #use t-approximation
```

```
##          2.5 %   97.5 %
## hours 1.498938 3.501062
```

```r
# estimate total and SE
svytotal(~hours,d0606)
```

```
##        total     SE
## hours 1617.5 233.28
```

```r
confint(svytotal(~hours,d0606),level=.95,df=4)
```

```
##          2.5 %   97.5 %
## hours 969.8132 2265.187
```

```r
##### Example 6.11

data(classpps)
nrow(classpps)
```

```
## [1] 20
```

```r
head(classpps)
```

```
##   class class_size finalweight hours
## 1     4         22       32.35   5.0
## 2     4         22       32.35   4.5
## 3     4         22       32.35   5.5
## 4     4         22       32.35   5.0
## 5    10         34       32.35   2.0
## 6    10         34       32.35   4.0
```

```r
d0611 <- svydesign(ids = ~class, weights=~classpps$finalweight, data = classpps)
d0611
```

```
## 1 - level Cluster Sampling design (with replacement)
## With (5) clusters.
## svydesign(ids = ~class, weights = ~classpps$finalweight, data = classpps)
```

```r
# estimate mean and SE
svymean(~hours,d0611)
```

```
##       mean     SE
## hours 3.45 0.4819
```

```r
confint(svymean(~hours,d0611),level=.95,df=4) #use t-approximation
```

```
##          2.5 %   97.5 %
## hours 2.112147 4.787853
```

```r
# estimate total and SE
svytotal(~hours,d0611)
```

```
##        total     SE
## hours 2232.2 311.76
```

```r
confint(svytotal(~hours,d0611),level=.95,df=4)
```

```
##          2.5 %   97.5 %
## hours 1366.559 3097.741
```

```r
##### Example 6.8

supermarket<-data.frame(store=c('A','B','C','D'),area=c(100,200,300,1000),
                        ti=c(11,20,24,245))
supermarket
```

```
##   store area  ti
## 1     A  100  11
## 2     B  200  20
## 3     C  300  24
## 4     D 1000 245
```

```r
supermarket$psi<-supermarket$area/sum(supermarket$area)
psii<-supermarket$area/sum(supermarket$area)
piik<- psii %*% t(psii/(1-psii)) + (psii/(1-psii)) %*% t(psii)
diag(piik)<-rep(0,4) # set the diagonal entries of the matrix equal to zero
piik  # joint inclusion probabilities
```

```
##            [,1]       [,2]       [,3]      [,4]
## [1,] 0.00000000 0.01726190 0.02692308 0.1458333
## [2,] 0.01726190 0.00000000 0.05563187 0.2976190
## [3,] 0.02692308 0.05563187 0.00000000 0.4567308
## [4,] 0.14583333 0.29761905 0.45673077 0.0000000
```

```r
pii<-apply(piik,2,sum)
pii # inclusion probabilities
```

```
## [1] 0.1900183 0.3705128 0.5392857 0.9001832
```

```
##### Example 6.9

supermarket2<-supermarket[3:4,]
supermarket2$pii <- pii[3:4] # these are the unit inclusion probs when n=2
jointprob<-piik[3:4,3:4]  # joint probability matrix for stores C and D
diag(jointprob)<-supermarket2$pii # set diagonal entries equal to pii
jointprob
```

```
##           [,1]      [,2]
## [1,] 0.5392857 0.4567308
## [2,] 0.4567308 0.9001832
```

```
# Horvitz-Thompson type
dht<- svydesign(id=~1,  fpc=~pii, data=supermarket2,
                  pps=ppsmat(jointprob),variance="HT")
dht
```

```
## Sparse-matrix design object:
##  svydesign(id = ~1, fpc = ~pii, data = supermarket2, pps = ppsmat(jointprob),
##     variance = "HT")
```

```
svytotal(~ti,dht)
```

```
##     total     SE
## ti 316.67 82.358
```

```
# Sen-Yates-Grundy type
dsyg<- svydesign(id=~1,  fpc=~pii, data=supermarket2,
                  pps=ppsmat(jointprob),variance="YG")
dsyg
```

```
## Sparse-matrix design object:
##  svydesign(id = ~1, fpc = ~pii, data = supermarket2, pps = ppsmat(jointprob),
##     variance = "YG")
```

```
svytotal(~ti,dsyg)
```

```
##     total     SE
## ti 316.67 57.094
```

```
##### Example 6.10

data(agpps)
jtprobag<-as.matrix(agpps[,20:34])
diag(jtprobag)<-agpps$SelectionProb
# Horvitz-Thompson type
dhtag<- svydesign(id=~1,  fpc=~SelectionProb, data=agpps,
                  pps=ppsmat(jtprobag),variance="HT")
svytotal(~acres92,dhtag)
```

```
##             total       SE
## acres92 936291172 70466858
```

```
# Sen-Yates-Grundy type
dsygag<- svydesign(id=~1,  fpc=~SelectionProb, data=agpps,
                   pps=ppsmat(jtprobag),variance="YG")
svytotal(~acres92,dsygag)
```

```
##             total       SE
## acres92 936291172 11715201
```

```
# Hartley-Rao approximation
sumsqprob<-sum(agpps$SelectionProb^2)/nrow(agpps)
dHRag<-svydesign(id=~1, fpc=~SelectionProb, data=agpps,
                   pps=HR(sumsqprob),variance="YG")
svytotal(~acres92,dHRag)
```

```
##             total       SE
## acres92 936291172 12148234
```

```
# With-replacement variance
dwrag<-svydesign(id=~1, weights=~SamplingWeight, data=agpps)
svytotal(~acres92,dwrag)
```

```
##             total       SE
## acres92 936291172 12293009
```