Combinations of clustering and stratification

Peter Lugtig

02 oktober, 2023

Because there is only a short lecture next week, there is a batch of exercises to be done this week:

i) The Hurvitz-Thompson estimator
ii) Other statistics than the mean
iii) Combinations of stratification and clustering

Exercise 1

This is an important exercise. It will introduce inclusion probabilities and design weights as two ways to work with the Horvitz-Thompson estimator (HT), which provides a flexible way to correctly specify your survey design, even when you can't work out the survey design (or when the survey design is very complex). To illustrate why the HT estimator is so useful, we will again use the 'boys' dataset

```
# install.packages("tidyverse")
# install.packages("magrittr")
# install.packages("survey")
# install.packages("sampling")

library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
## Warning: package 'ggplot2' was built under R version 4.0.5
## Warning: package 'tibble' was built under R version 4.0.5
## Warning: package 'tidyr' was built under R version 4.0.5
## Warning: package 'readr' was built under R version 4.0.5
## Warning: package 'dplyr' was built under R version 4.0.5
## Warning: package 'stringr' was built under R version 4.0.5
## Warning: package 'forcats' was built under R version 4.0.5
```

```
library(magrittr)
```

```
## Warning: package 'magrittr' was built under R version 4.0.5
```

```
library(survey)
```

```
## Warning: package 'survey' was built under R version 4.0.5
## Warning: package 'Matrix' was built under R version 4.0.5
## Warning: package 'survival' was built under R version 4.0.5
```

```
library(sampling)
```

```
## Warning: package 'sampling' was built under R version 4.0.5
```

```
boys <- read_rds("boys.RDS")
```

Working with probabilities & design weights

Whenever we use unequal selection probabilities (e.g. cluster and stratification) we need to specify a 'svydesign()' object that tells R the structure of our sampling design, so that R can compute our outcome statistics per stratum/cluster and then combine them.

There is an entirely different way to work with datasets that have used unequal sampling designs: the Hurvitz Thompson (HT)-estimator uses the individual inclusion probabilities of every case in the dataset.

We will for now use a part of the cluster sampling design we used in week 4 where we draw a two-stage cluster sample. The code below is a copy of the code you have used before (so you may also just open your code, and run that)

We first draw 2 towns SRS, and then in each town 75 cases using SRS again using the code below. Feel free to run the code with another sampling design (e.g. more or fewer). It is good to practice with this, and a good way is to just alter the basic code I provide below.

```
# first the sampling bit (repeated from week 40)
boys$id <- 1:nrow(boys)
table(boys$town)
```

```
##
##   1   2   3   4   5
## 191  81 239 161  73
```

```
samplesizestown <- c(table(boys$town))

set.seed(123)
samplec <- sample(5,2)
samplec
```

```
## [1] 3 2
```

```
# removing missing values in town
boys2 <- boys %>%
    filter(!is.na(town))%>%
    arrange(town)
# draw the cluster sample
clustersample <-
    boys2 %>%
    filter(town %in% samplec) %>%
    group_by(town) %>%
    filter(srswor(75, n()) == 1)

# Now we specify the svydesign
# specify the fpc, which now exists of two components:
# the number of clusters, and the size within every sampled cluster
clustersample$ncluster <- 5 # 5 towns in population
clustersample$clustersize <- NA # empty first, then add sample sizes in towns 2 and 3
clustersample$clustersize[clustersample$town == 2] <- nrow(boys2[boys2$town == 2, ])
clustersample$clustersize[clustersample$town == 3] <- nrow(boys2[boys2$town == 3, ])

# or in a tidy way
```

```
# clustersample <-
#     boys2 %>%
#     group_by(town) %>%
#     summarize(clustersize = n()) %>%
#     right_join(clustersample1)

# and specify the cluster design
clusterdesign <- svydesign(id = ~town + id,
                           fpc = ~ncluster + clustersize,
                           data = clustersample)
```

Q1

What is the mean for the variable 'wgt' and its standard error?

The Hurvitz Thompson estimator

Because we have drawn the clusters using SRS, the inclusion probabilities of the individual boys in the dataset vary across towns. Children from smaller towns have a higher probability of being selected into the sample. By specifying the svydesign() object, we tell R to compute the mean separately in every cluster, and then combine the means from the clusters weighted by the cluster size. Can you compute for the individuals in clusters 2 and 3 (within the clusters drawn) what was the inclusion probability?

The probabilities can also be derived from the clusterdesign svydesign object by asking for:

```
clusterdesign[["prob"]]
# exactly the same as manually computed ones
# just asking for:
clusterdesign
# actually also tells us what design we just specified!
# handy!
```

The central idea of the HT estimator is that instead of specifying the specific clustering and stratification that we have in our sample, we can do a much easier job by simply including the inclusion probability for every case in our svydesign() object. Statistics like the mean can be easily computed when the inverse of the inclusion probability is used as a weighting factor for every individual in the dataset. Rather than computing the mean per stratum/cluster, the HT estimator uses the inclusion probabilities (p) of every individual (i) in the sampling design directly: pi.

Using the HT estimator in R is easy. We simply use the inclusion probabilities in the probs=~ command. See the code below.

##Q2: What is the survey mean for the variable 'wgt' and its standard error under the HT estimator?

```
# HT estimator
clustersample$fpc <- 748
HT1<- svydesign(id=~town, probs =~prob, data=clustersample)
```

You should find here that the HT estimator results in a larger standard error than when specifying the cluster correctly (!). Why is this? Obviously, something happens when we compute the variance using the HT estimator. We will return to this in a bit.

Design weights

Often, public use datasets (like your adopted survey) do not include the specific cluster or stratification indicators, nor the inclusion probabilities, but rather "Design weights". A Design weight is an indicator that tells you "how many population elements" a row in your dataset represents in the population, and is the

inverse of the inclusion probability (pi). Rather than using probabilities in the probs=~ statement, we can also use design weights r in our svydesign object as a HT estimator.

In summary: as soon as we provide either probabilities or weights in the 'weights=~ command, the survey package will switch to use the HT estimator.

##Q3: Check the code below. Do you get the same results using inclusion probabilities and weights?

```
clustersample$designweight <- 1/clustersample$prob
HT2 <- svydesign(id=~town,weights=~designweight,data=clustersample)
svymean(~wgt,design=HT2,deff=T,na.rm=T)
# equivalent!
```

The tricky estimation of variances under HT-estimators

Stuart (1984) showed you that the variance of a cluster or stratified design can be computed as the weighted average of variance in every stratum/cluster. Hurvitz-Thompson works with individual inclusion probabilities, so how do we define the variance now?!

This course will not discuss methods to estimate the variance (of e.g. the mean) under all conditions in detail, but below I will show some ways to compute a variance for HT estimators.

There are over 50 different algorithms to compute the variance for HT type estimators, and there is no agreement over which method works (generally) better.

The most common way to compute the variance is by Taylor series expansion (review your slides from Fundamentals on how this works). The idea here is that we can approximate the variance of an unknown distribution (the inclusion probabilities) by fitting a series of polynomials (x, X^2, X^3, X^4, etc.), and then derive the total variance by averaging the variance over the fitted function.

Another (perhaps more modern) way is to use Bootstrapping. You will learn more about this later this semester, but the general idea here is that you can resample (with replacement!) from the actual sample to get a large new number of samples (e.g 1000 new replicate samples). Remember we did this for simple random sampling: we can estimate the empirical variance of the sampling distribution simply by repeating our sampling procedure X times (say 1000).

After this, you get 1000 point estimates for every individual statistic (the mean in this case), which you can then use to estimate the variance around the mean. This also means that you get 1000 inclusion probabilities and design weights per case. The survey package can however work with such "bootstrap replicate weights", and pool the estimates for you.

If you want to know more about different versions of the Taylor expansions available, the survey package includes a few standard algorithms (see below). If you would like to read more (this will not be tested), you may read:

Mattei & Tille (2005) Evaluation of Variance Approximations and Estimators in Maximum Entropy Sampling with Unequal Probability and Fixed Sample Size, Journal of Official Statistics, Vol. 21, No. 4, 2005, pp. 543–570

The differences between these algorithms is usually quite small.

bonus. Run the code below and see how HT variance estimator differ slightly

Look at the size of the variances. Investigate the size of the differences. Are these large or small?

```
clustersampleNA <- subset(clustersample, !is.na(wgt))
HT3 <- svydesign(id=~town,  probs=~prob, data=clustersampleNA, pps="brewer")
HT4 <- svydesign(id=~town,  probs=~prob, data=clustersampleNA, pps="overton")
HT5 <- svydesign(id=~town,  probs=~prob, data=clustersampleNA, variance="HT")
```

```r
svymean(~wgt,design=HT3,deff=T,na.rm=T) #deff=5.67 # equivalent
```

```
##        mean      SE   DEff
## wgt 41.5358  4.6426 5.6713
```

```r
svymean(~wgt,design=HT4,deff=T,na.rm=T) #deff=4.27 # some underestimation
```

```
##        mean      SE   DEff
## wgt 41.5358  4.0261 4.2652
```

```r
svymean(~wgt,design=HT5,deff=T,na.rm=T) #deff=5.67 # equivalent
```

```
##        mean      SE   DEff
## wgt 41.5358  4.6426 5.6713
```

```r
# all designs overestimate the variance compared to the normal cluster design
# this is quite common with the HT estimator: it is 'conservative' in estimating standard errors
```

(bonus) why use a HT estimator if variance estimation is so tricky?

In a simple sample design, you don't need to use the HT estimator. But in real life sampling designs are complex. For example, you could in real life situations encounter situations where you combine stratification and clustering, resulting in some really small cell sizes, and others large. In such situations, you will have very different group sizes, and quite likely end up with subgroups of just a few or even 1 case. Computing variances for such small groups is risky (as the variance can become very small or even 0), and if you somehow end up with just 1 case in a cluster/stratum, the variance is undefined!

The example below illustrates this. We first add a bit of random noise to the inclusion probabilities, so that we get group sizes of 1 (and variances using methods covered in earlier weeks cannot be calculated)

```r
# add some noise to the inclusion probabilities
require(MASS)
```

```
## Loading required package: MASS
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```r
clustersample$prob2 <- rnorm(n=nrow(clustersample),mean=clustersample$prob,sd=.02)
table(clustersample$prob2) # all diifferent probabilities now.
# HT estimator
HT6 <- svydesign(id=~town, probs=~prob2, data=clustersample)
svymean(~wgt,design=HT6,deff=T,na.rm=T) #deff=5.46 # small difference, due to small diff in standard er
```

The main take-away of all of this? The HT estimator is flexible, and easy to use under any sampling design. You however give

Exercise 2

This week, we will discuss combinations of stratification and clustering. Lets first revisit stratification and clustering as covered last week by revisiting the final questions of those exercises.

Clustering (Q4 last week but with slightly different numbers).

Load the "boys.R" dataset. You can either optimize your cluster-design based on costs or precision, or balance both. Imagine it costs 10 euros to conduct one interview (variable costs).

However, it also costs 500 euros to conduct interviews in one cluster (fixed costs per cluster).

An SRS of size 100 will in this case normally cost you "100 * 10 + 5 *500 = 3500 euros.

The cluster design specified in question 3 will cost you "100 * 10 + 2 * 500 = 2000 euros.

You now have to balance costs and precision. Assume that you have a fixed budget of 2500 euros. Doing an SRS is here not an option, because you would spend all your money on fixed costs. At the other extreme, you can sample in just 1 cluster and interview 400 cases.

Question 1:

Can you work out, using the variable 'wgt' (weight) as your dependent variable, what would be the right number of clusters to draw?

And then compute the mean in 'wgt' for the different cluster designs

Question 2

In practice, we may not want to draw clusters (PSUs) using an SRS, or Proportional to Size (PPS - as we have done so far), but we may want to stratify *before* drawing the clusters. In order to understand this: Imagine we are interested in doing a survey among ethnic minorities, which - at least in many countries - are to be found in the big cities. We therefore want to increase the probability of drawing cities in our sample.

Thinking about the 'wgt' variable - that we are lucky to know for our entire population - do you think it makes sense to stratify at the PSU level? And how is this for the variable 'hgt'?

Question 3

Assume (see question 1) that because we want to balance costs and precision, we want to sample 2 clusters. We have so far used sampling proportional to size for the cluster sizes. Can we stratify on 'wgt' in the first step when we draw PSUs. How would we do this? (you may take a sneak peak at .RMD file to see how I solved this using Neyman allocation at the PSU level)

Below you can find code where we adjust cluster probabilities based on neyman allocation criteria. Run the code, and work out line-by-line what happens.

Question 4:

what happens to the s.e. and precision compared to the earlier design where you drew 2 clusters?

Finally, we can take a cluster sample like we have done before, but then within every PSU stratify the sample (possibly on a different variable) before taking a sample.

Question 5:

What about truly combining clustering (to save costs) with stratification (to optimize precision?): We can combine them. For example:

Take a clustersample proportional to size with 2 PSUs, and then stratify the sample on the variable 'agecat' using neyman allocation for example.

In summary: we can do the stratification at two moments in our sampling procedure: 1. We can stratify using population data information (like we did last week), then draw the clusters (PPS), and draw the stratified sample within each of the clusters in the same way. 2. We can first draw the cluster sample, and then within each cluster figure out what would be a good way to stratify. This could lead to a design where we use different allocation proportions in each cluster, or even different variables in different clusters! Such a design sounds strange, but it is commonly used in multi-country surveys, where in one country you may have information on your sampling frame about age, and in the other country you don't.

For this exercise, use approach 1. First draw 2 clusters, and then draw a stratified sample within every cluster based on the variable 'agecat'. Note that in some cases you may run into the issue that there is only one PSU for a stratum (e.g. when you stratify on towns). In that case the option

```
options(survey.lonely.psu="adjust")
```

will tell R how to deal with this. The method "adjust" ensures that the stratum contribution to the variance is taken to be the average of all the strata with more than one PSU. This implies this PSU does not contribute to the variance. For more info, see http://r-survey.r-forge.r-project.org/survey/exmample-lonely.html

Question 6 (optional - no answer available):

If you feel like practicing more: You can try to see whether you can improve precision by stratifying within the clusters in a different way. Use the same two clusters. Figure out how you could stratify for each cluster separately. Compare your answer to question 4. What do you find?

Question 7 (optional)

Can you compute the inclusion probabilities for the design you specified in question 5, and use HT-estimation (use the probs= or weights= command rather than the fpc)?

Exercise 3

So far, I have always asked you to compute the mean as a statistic. The mean is a very useful statistic, as many concepts in statistics can be expressed as means. For example, a regression model including variables X and Y, can be expressed as a set of link-functions that describe the relation between values of x and the conditional mean (or predicted mean) in y.

Still, you may in practice want to compute other things than the mean. The survey package includes many different functions, which I will not show all, but I will show you some functions that may be useful. The idea of this exercise is that you simply run the code first, and then change some variables to test and see what every option does.

Let's first reload our example dataset.

```
boys <- readRDS("boys.RDS")
boys <-
    boys %>%
    mutate(agecat = cut(age,
                        breaks = c(0, 1, 5, 10, 22),
                        labels = c("Younger than 1",
                                   "1 - 5",
                                   "5 - 10",
                                   "Older than 10")))
```

#Creating a sample, and survey design object

Below I will draw a simple SRS of size 200, like you did in week 3. You can of course also use a cluster or stratified sample

```
# SRS of 200
set.seed(11)
boys$srs <- srswor(200,nrow(boys))

srs <- subset(boys,srs==1)
srs$fpc <- nrow(boys)

# now with fpc - this is correct!
srsdesign <- svydesign(id=~1,fpc=~fpc,data=srs)
svymean(~wgt,na.rm=T,design=srsdesign,deff=T) # you can use a different y-variable
```

#describing the dependent variable Now, lets ask for some statistics for the 'wgt' income.

Question 1.

a. Inspect the output: do you understand what is going on?
b. Can you ask for the 50th quantile (the median), or the 95th?

```
# can we also compute other things? Sure!For example, the total weight.
# In the context of this dataset it makes no sense, but if you do a business survey,
# the total may show the total revenue, or total production for example.
sum(boys$wgt,na.rm=T) #
svytotal(~wgt,na.rm=T,design=srsdesign,deff=T)
# get estimates of weighted percentiles (here the 25th )
svyquantile(~wgt,na.rm=T, srsdesign, quantiles = 0.25, ci = TRUE)
```

#Saving the results as an object

You can also ask for multiple means, and save these as an object. This has other benefits, because you can

do calculations on this object, e.g. get a Confidence Interval

Question 2.

Run the code below, and inspect how the output looks like

```
mysvymean <- svymean(~wgt+hgt, na.rm=T, srsdesign, deff = TRUE)
coef(mysvymean)
SE(mysvymean)
deff(mysvymean)
# and get a confidence interval
confint(mysvymean)
```

finally, bi- and multivariate relations Perhaps you are more interested in computing cross-tables or regression-coefficients. This can also be done using the functions below.

```
#we can estimate crosstables, and t-tests
svytable(~town+agecat,design=srsdesign) # population tables
svyby(~wgt, by=~agecat,svymean,design=srsdesign)

# we can extend to do a chi.sq test
svychisq(~town+agecat, srsdesign, statistic="adjWald")

# or regression
svyglm(wgt~hgt+age+town, design=srsdesign)
# and we can access parts of the output
coef(svyglm(wgt~hgt+age+town, design=srsdesign))
```

Question 3 (no answer available):

Can you rerunyour code using a stratified or cluster sample (you may choose a design from week 5 or 6 or specify a new design), but focus on a relation between variables (choose some relationship you may find interesting?

– End of File –