Wednesday, Nov 20

Survey Weights Revisited

Estimators of τ can often be written as

$$\hat{\tau} = \sum_{i \in \mathcal{S}} w_i y_i,$$

where w_i is the *survey weight* for the i-th element in the sample. Where does w_i come from?

1. The sampling design. If there is no adjustment/re-weighting (see below) then $w_i = 1/\pi_i$ where π_i is the inclusion probability of the i-th element. See this lecture on inclusion probabilities.
2. Calibration. Certain estimators such as ratio and (generalized) regression estimators effectively adjust or re-weight the weights based on known totals for one or more auxiliary variables. This also includes post-stratification and raking. See this lecture on how weights on re-weighting and calibration.
3. Non-response/detectability. Further adjustment or re-weighting may be done to account for known (or estimated) probabilities of response or detection. See the lectures on detectability and non-response.

Estimation of μ

There are a couple of ways that we can estimate μ using survey weights. One is

$$\hat{\mu} = \frac{\sum_{i \in \mathcal{S}} w_i y_i}{\text{number of elements in the population}},$$

if the denominator is *known*. If not, then we can use

$$\hat{\mu} = \frac{\sum_{i \in \mathcal{S}} w_i y_i}{\sum_{i \in \mathcal{S}} w_i},$$

since $\sum_{i \in \mathcal{S}} w_i$ is an estimator of the number of elements in the population.

Target Variable Specification

Once the weights are known, the user can define y_i depending on what they want to estimate.

1. y_i can be the value of some target variable of interest.
2. y_i can be defined as equal to one for all elements in the population. Then $\hat{\tau} = \sum_{i \in \mathcal{S}} w_i y_i = \sum_{i \in \mathcal{S}} w_i$ is an estimator of the number of elements in the population.
3. y_i can be defined as

   $$y_i = \begin{cases} 1, & \text{if the } i\text{-th element is in a given domain,} \\ 0, & \text{otherwise,} \end{cases}$$

   so that $\hat{\tau}$ is an estimator of the *number of elements in the domain* and $\hat{\mu}$ is an estimator of the *proportion of elements in the domain*.
4. y_i can be defined as

   $$y_i = \begin{cases} y_i, & \text{if the } i\text{-th element is in a given domain,} \\ 0, & \text{otherwise,} \end{cases}$$

   so that $\hat{\tau}$ is an estimator of the total for *the sub-population of elements in the domain* and $\hat{\mu}$ is an estimator of the mean for *the sub-population of elements in the domain*.

Computational Convenience

The advantage of survey weights is that they make it relatively easy for non-specialists to compute estimates using only the survey weights and the (specified) target variable values. (However computing the (estimated) variance of an estimator to compute the bound on the error of estimation still requires more information and expertise.)

Example: The following table shows the value of the target variable and the corresponding survey weights for a sample of five elements obtained using some unknown probability sampling design.

Target Variable	Survey Weight
3	17.2
9	18.8
5	27.8
5	20.4
4	20.6

How do we compute estimates of τ and μ using only this information?

R Demonstration of Using Survey Weights

Here I will demonstrate the use of sampling weights from the European Social Survey (ESS). Survey data from the ESS are available for download. For this example we will use data from the ninth round of the survey that was conducted from 2018 to 2020.

```
essdata <- read.csv("ESS9e03_2.csv")
```

You can download the data yourself. Registration is required but is free and not restrictive.

The data require a little bit of formatting before we can use it. Here I re-code the missing responses, select the variables we want to use (just to keep the data to a manageable size), and drop elements with missing data. I am also going to create some age groups based on the age of the respondent. This can be done many different ways in R. I will use the **dplyr** and **tidyr** packages.

```
library(dplyr)
library(tidyr)
ukdata <- essdata %>% filter(cntry == "GB") %>%
  select(psu, stratum, dweight, pspwght, agea, hmsacld) %>%
  mutate(agea = ifelse(agea == 999, NA, agea)) %>%
  mutate(hmsacld = ifelse(hmsacld %in% c(7,8,9), NA, hmsacld)) %>%
  drop_na() %>%
  mutate(agegroup = cut(agea, breaks = c(min(agea) - 1,
    quantile(agea, c(0.25, 0.5, 0.75)), max(agea))))
```

Here you can see the first 20 responses.

```
head(ukdata, 20)
```

```
     psu stratum   dweight   pspwght agea hmsacld agegroup
1  12304    1588 2.0375460 2.5500326   19       1  (14,37]
2  12445    1440 1.0187730 1.0932655   42       1  (37,53]
3  12337    1560 1.5281595 1.9591819   15       1  (14,37]
4  12255    1554 0.5093865 0.4731422   66       1  (53,67]
5  12401    1417 1.0187730 1.2063588   52       2  (37,53]
6  12330    1558 1.0187730 1.3198249   21       2  (14,37]
7  12147    1523 2.0375460 1.4712680   76       4  (67,90]
8  12392    1402 0.5093865 0.4618744   84       3  (67,90]
9  12161     1478 0.5093865 0.6377798   38        1  (37,53]
10 12446     1444 1.0187730 0.7758576   64        2  (53,67]
11 12242     1560 1.5281595 1.7110010   46        2  (37,53]
12 12233     1539 0.5093865 0.4552839   35        2  (14,37]
13 12309     1576 0.5093865 0.4271142   63        2  (53,67]
14 12331     1488 0.5093865 0.6530607   25        1  (14,37]
15 12376     1585 1.0187730 0.8542284   62        1  (53,67]
16 12388     1589 0.5093865 0.3951095   82        2  (67,90]
17 12133     1452 0.5093865 0.4164525   73        4  (67,90]
18 12396     1413 1.0187730 1.2063588   40        1  (37,53]
19 12207     1527 1.0187730 1.0036734   70        2  (67,90]
20 12143     1544 1.0187730 0.8849896   62        4  (53,67]
```

The variables `psu` and `stratum` identify the primary sampling unit and the stratum. I believe the survey uses a kind of stratified multi-stage cluster sampling design. The variables `dweight` and `pspwght` are the design weight and survey weight, respectively. The latter has been adjusted through a re-weighting method (post-stratification) using raking with age, gender, education, and region as the auxiliary variables. The variable `agea` is the age of the respondent, which I have used to form four age groups (`agegroup`) based on quartiles. The target variable for this demonstration will be `hmsacld` which is a response to the statement "Gay male and lesbian couples should have the same rights to adopt children as straight couples." The response was on a 5-point scale: agree strongly (1), agree (2), neither agree nor disagree (3), disagree (4), disagree strongly (5).

To estimate the mean response for the population, we simply need to multiply the target variable (`hmsacld`) by the survey weight (`pspwght`), add these up across the sampled elements, and divide by the sum of weights for the sampled elements. The formula is

$$\hat{\mu} = \frac{\sum_{i \in \mathcal{S}} w_i y_i}{\sum_{i \in \mathcal{S}} w_i}$$

where y_i is the target variable (`hmscald`) and w_i is the survey weight (`pspwght`). Here's how we could do this in R.

```r
ukdata %>% summarize(y = sum(hmsacld * pspwght) / sum(pspwght))
```

```
         y
1 2.194636
```

We can also estimate the mean response for each age group by doing this separately for each part of the sample. Technically we can do this by defining y_i as equal to zero if the elements is *not* in the domain of interest, but we can also do this simply by doing the calculation separately for each domain.

```r
ukdata %>% group_by(agegroup) %>% summarize(y = sum(hmsacld * pspwght) / sum(pspwght))
```

```
# A tibble: 4 x 2
  agegroup     y
  <fct>    <dbl>
1 (14,37]   1.76
2 (37,53]   2.10
3 (53,67]   2.39
4 (67,90]   2.96
```

Note that this did not require any knowledge of the sampling design and the calibration. It also does not require sophisticated software. Everything that was done could be done with any statistical package or a spreadsheet program. But we can confirm that the estimates are consistent with what would be obtained using specialized software like the **survey** package for R.

```r
library(survey)
ukdesign <- svydesign(ids = ~psu, strata = ~stratum, weights = ~pspwght, data = ukdata)
```

```
svymean(~hmsacld, design = ukdesign)
```

```
          mean     SE
hmsacld 2.1946 0.0338
```

```
svyby(~hmsacld, by = ~agegroup, design = ukdesign, FUN = svymean)
```

```
        agegroup  hmsacld         se
(14,37]  (14,37] 1.760756 0.05010013
(37,53]  (37,53] 2.100307 0.06365683
(53,67]  (53,67] 2.390973 0.05664507
(67,90]  (67,90] 2.959859 0.05751275
```

Same estimates (aside from differences in rounding), but the **survey** package is also capable of estimating standard errors *given correct information about the sampling design.* Here that information is communicated through the identity of the primary sampling units, the identity of the strata, as well as the weights.

We can also treat `hmscald` as a categorical variable if we want to estimate the proportion of people in the population that would respond in a certain way.

```
ukdata <- ukdata %>%
  mutate(hmsacld = factor(hmsacld, levels = 1:5,
    labels = c("strongly agree", "agree", "neither",
      "disagree", "disagree strongly")))
ukdesign <- svydesign(ids = ~psu, strata = ~stratum, weights = ~pspwght, data = ukdata)
svymean(~hmsacld, design = ukdesign)
```

```
                             mean     SE
hmsacldstrongly agree    0.338782 0.0135
hmsacldagree             0.342761 0.0114
hmsacldneither           0.158160 0.0101
hmsaclddisagree          0.105632 0.0082
hmsaclddisagree strongly 0.054665 0.0063
```

```
confint(svymean(~hmsacld, design = ukdesign))
```

```
                              2.5 %     97.5 %
hmsacldstrongly agree    0.31228061 0.36528393
hmsacldagree             0.32044338 0.36507824
hmsacldneither           0.13845528 0.17786513
hmsaclddisagree          0.08963606 0.12162831
hmsaclddisagree strongly 0.04225585 0.06707323
```