Sampling Part 1

Will Doyle

```
. capture log close                          // closes any logs, should they be open

. set linesize 90

. log using "sampling_part1.log", replace    // open new log
----------------------------------------------------------------------------------------------
      name:  <unnamed>
       log:  /Users/doylewr/lpo_prac/lessons/s1-06-sampling/sampling_part1.log
  log type:  text
 opened on:  30 Sep 2020, 10:28:19
```

PURPOSE

In most stats classes, all samples are assumed to be simple random samples from the population, with each unit having exactly the same probability of being selected. In practice, this is extremely rare. Samples are usually designed with unequal probabilities of selection across different groups. Because survey methodology is complex, this lecture cannot pretend to be comprehensive. Instead, it is meant to expose you to various sampling designs often found in education research as well as the formulas for computing means and variances of some of the simpler designs.

```
. clear all                                  // clear memory


. set more off                               // turn off annoying "__more__" feature

. global datadir "./"
```

Simple random sampling (SRS)

Formulas

Where y_i is value of y for the ith unit:

Sample mean

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

Sample variance

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

Standard error of sample mean

$$\bar{y}_{se} = \sqrt{\frac{s^2}{n}}$$

```
. use ${datadir}fakesat, clear

. egen scoretot = total(score)           // total of all scores

. scalar popmean = scoretot / _N         // population mean score

. gen sqdiff = (score - popmean)^2       // (Xi - Xbar)^2

. egen sst = total(sqdiff)               // total of squared differences

. scalar popvar = sst / (_N - 1)         // population variance (not super pop)

. scalar popsd = sqrt(popvar)            // population standard deviation

. scalar popsem = popsd / sqrt(_N)       // standard error of population mean

. scalar list popmean popsd popsem
   popmean =  499.87537
     popsd =  99.859033
    popsem =  .08153456

. summarize score

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
       score |  1,500,000    499.8754    99.85903        200        800

. mean score

Mean estimation                   Number of obs   =  1,500,000

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
       score |   499.8754   .0815346      499.7155    500.0352
```

```
--------------------------------------------------------------

. keep score
```

Compute SRS mean and variances

Now we'll take a simple random sample (SRS) of 10% of our test takers and compute our statistics.

```
. sample 10
(1,350,000 observations deleted)

. egen scoretot = total(score)            // total of all scores

. scalar sampmean = scoretot / _N         // sample mean score

. gen sqdiff = (score - sampmean)^2       // (Xi - Xbar)^2

. egen sst = total(sqdiff)                // total of squared differences

. scalar sampvar = sst / (_N - 1)         // sample variance

. scalar sampsd = sqrt(sampvar)           // sample standard deviation

. scalar sampsem = sampsd / sqrt(_N)      // standard error of sample mean

. scalar list sampmean sampsd sampsem
  sampmean =  499.60491
    sampsd =   99.73197
   sampsem =  .25750684

. summarize score

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
       score |    150,000    499.6049    99.73197        200        800

. mean score

Mean estimation                   Number of obs   =    150,000

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
       score |   499.6049   .2575068      499.1002    500.1096
```

```
------------------------------------------------------------------
```

As you can see, our sample mean, variance, and standard error of the mean are about the same as the population values. $\bar{y}_{se}$ is a little higher, which is to be expected since we are basing our estimate off fewer observations. And in both cases, our hand calculations are the same as those given by Stata. That is always a good sign!

Description and formula

Consider the normal estimate of the standard error of the mean, show in equation 3 above. In cases where the proportion of the population that is sampled is quite large, this will in fact be an overestimate of the standard error of the mean. This is because in classical statistical theory, the sample is conceived as being from an infinitely large population. The finite population correction is a way of adjusting for the fact that the sample actually may be more representative than standard approaches would suggest. The finite population correction (FPC) is calculated as:

$$fpc = \sqrt{\frac{N-n}{N-1}}$$

where N is the population size and n is the sample size. As you can see, as n grows small relative to N, the FPC will approach 1 and the correction will be very small. As n becomes a larger fraction of N, the opposite is true. The FPC is rarely used in practice, but it should be used whenever the population size is known. Calculating $\bar{y}_{se}$using the FPC is done as follows:

$$\bar{y}_{se} = \sqrt{\frac{s^2}{n}} \times (fpc) = \sqrt{\frac{s^2}{n}}\sqrt{\frac{N-n}{N-1}}$$

Example

We'll again use some fake data to test our formulas. This time we have test score data for 50 students from a single large class. Let's say, for some mysterious reason, we only have access to information from 30 students. Maybe we did an exit poll of grades after class and assume that the 30 responses represent a random sample (highly unlikely, but we'll go with it for now). This number of students represents a sizeable portion of the population so we should adjust our estimate of the error the average score to take that into account.

```
. use ${datadir}singleclasstest, clear

. sum score
```

```
    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
       score |         50    80.02087    5.986087   64.91638   93.89526

. scalar N = _N

. sample 30, count
(20 observations deleted)

. scalar n = _N

. egen scoretot = total(score)           // total of all scores

. scalar xbar = scoretot / n             // sample mean score

. gen sqdiff = (score - xbar)^2          // (Xi - Xbar)^2

. egen sst = total(sqdiff)               // total of squared differences

. scalar var_x = sst / (n - 1)           // sample variance

. scalar sd_x = sqrt(var_x)              // sample standard deviation

. scalar sampsem = sd_x / sqrt(n)        // standard error of sample mean

. scalar list xbar var_x sd_x sampsem
      xbar =  79.749056
     var_x =  35.703024
      sd_x =  5.9752007
   sampsem =  1.0909174

. scalar fpc = sqrt((N - n) / (N - 1))

. scalar sampsemfpc = sampsem * fpc

. scalar list sampsem sampsemfpc
   sampsem =  1.0909174
sampsemfpc =  .69696157

. di xbar - invnormal(.975) * sampsem
77.610897

. di xbar + invnormal(.975) * sampsem
81.887215

. di xbar - invnormal(.975) * sampsemfpc
```

```
78.383036

. di xbar + invnormal(.975) * sampsemfpc
81.115076
```

Comparing the 95% confidence intervals, we can see they are a little tighter when the FPC is used. This is a reflection of our knowledge that our sample respresents a sizeable portion of the population and therefore is a better estimate than the standard formula will compute.

SRS and frequency weights

Sometimes data are reported in what's known as a frequency-weighted design. In such a setup, observations that take on the same values are reported only once, with a weight that is equal to how many times this particular set of observations was reported. This was a very common way of formatting data when computer memory was expensive, but less common now. You may still run across it from time to time so it's good to know about.

To demonstrate, we'll again use the fake SAT data. This time, however, the data are in a frequency format. Using the weight option for the `mean` command, we can set `[fw = freq]` and get the same estimates for mean score as we did with the full dataset.

```
. use ${datadir}fakesat_freq, clear

. list if _n < 11

     +--------------+
     | score   freq |
     |--------------|
  1. |   200   2412 |
  2. |   210    908 |
  3. |   220   1213 |
  4. |   230   1539 |
  5. |   240   2045 |
     |--------------|
  6. |   250   2725 |
  7. |   260   3313 |
  8. |   270   4305 |
  9. |   280   5393 |
 10. |   290   6621 |
     +--------------+

. mean score
```

```
Mean estimation                   Number of obs   =         61

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
       score |        500    22.7303      454.5326    545.4674
--------------------------------------------------------------

. mean score [fw = freq]

Mean estimation                   Number of obs   =  1,500,000

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
       score |   499.8754   .0815346      499.7155    500.0352
--------------------------------------------------------------
```

SRS with inverse probability weights

For most survey-based social science data, it is unlikely that all population members have the same probability of being sampled. This is a problem when the probability of selection is correlated with the quantities we hope to estimate. Without accounting for the probability of selection, our estimates will be biased, perhaps severely.

Going back to our fake SAT data, let's assume that our sample data come from voluntary responses and that test takers are more likely to report their scores if those scores are high (a not unreasonable situation). For purposes of the example, let's assume that we know the probability that a test taker will select to report his or her results (something that we are generally *very* unlikely to know). We generate our 1% sample this time based on the probability of reporting and check the unadjusted sample mean.

```
. use ${datadir}fakesat, clear

. gen preport = score / 1000 + .1 * (score / 10000)^2 + rnormal(0, .025)

. gsample 1 [w = preport], percent
(91 observations created)
(1,485,091 observations deleted)

. mean score

Mean estimation                   Number of obs   =     15,000
```

```
------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+----------------------------------------------
       score |    520.028   .8028213      518.4544    521.6016
------------------------------------------------------------
```

As expected the mean score is much higher than the population average (which should be around 500). Since we are omniscient researchers, we can generate inverse probability weights using the probability of reporting. Thinking it through, these weights will downweight those with a high likelihood of reporting and upweight those with a low likelihood, hopefully improving our estimate of the population mean score in the process.

```
. gen pweight = 1 / preport

. mean score [pweight = pweight]

Mean estimation                   Number of obs   =     15,000

------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+----------------------------------------------
       score |   499.8853   .8738054      498.1726    501.5981
------------------------------------------------------------
```

Description

Stratified sampling is a widely used and broadly applicable way of designing a sample. In stratified sampling, a set of strata are selected from the population, then samples are taken from within each strata. An example would be taking a sample of students from within elementary, junior, and high schools, with level of school as the strata. The idea is that strata are different in some fundamental way from each other but internally similar. Strata should effectively partition the population space, that is, not overlap and fully account for the population when put together.

Formulas

The notation for this type of sampling design is as follows:

stratum mean

$$\bar{y}_h = \frac{1}{N_h}\sum_{j=1}^{N_h} y_{hj}$$

stratum variance

$$s_h^2 = \frac{1}{N_h - 1}\sum_{j=1}^{N_h}(y_{hj} - \bar{y}_h)^2$$

population mean

$$\bar{y} = \frac{1}{N}\sum_{h=1}^{L} N_h \bar{y}_h$$

population mean variance

$$s^2 = \sum_{h=1}^{L}\left(\frac{N_h}{N}\right)^2\left(\frac{N_h - n_h}{N_h - 1}\right)\left(\frac{s_h^2}{n_h}\right)$$

where

- N is the population total
- y_{hj} is observation j within stratum h
- $\bar{y}_h$ is the mean within stratum h
- $ N_h$ is the total number within stratum h
- n_h is the number sampled within stratum h
- $s_h^2) is the variance within stratum h

Example

This time we'll using fake data on a high school with grades 9-12. A student in this school is considered to be *at risk* if his or her test score falls below a certain cut off, commensurate with the student's grade. Looking at the administrative data we can see the proportion at risk, within each grade and across the school.

```
. insheet using ${datadir}fakehs.csv, clear
(8 vars, 2,003 obs)

. scalar stupop = _N

. mean atrisk                                    // overall
```

```
Mean estimation                   Number of obs   =      2,003

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
      atrisk |    .337993   .0105719      .3172599    .3587261
--------------------------------------------------------------

. mean atrisk, over(grade)                 // within each grade

Mean estimation                   Number of obs   =      2,003

            9: grade = 9
           10: grade = 10
           11: grade = 11
           12: grade = 12

--------------------------------------------------------------
        Over |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
atrisk       |
           9 |   .3252336   .0202723      .2854766    .3649907
          10 |   .3089431   .0208524      .2680485    .3498377
          11 |   .3677932   .0215218      .3255857    .4100008
          12 |   .3509514    .021968      .3078688     .394034
--------------------------------------------------------------

. global ss = 50                           // set within grade sample size

. sample $ss, count by(grade)              // sample
(1,803 observations deleted)

. preserve

. collapse (mean) propatr = atrisk (sd) sdatr = atrisk (first) nstgrade, by(grade)

. scalar Ybar9 = propatr[1]                // 9th grade average

. scalar Ybar10 = propatr[2]               // 10th grade average

. scalar Ybar11 = propatr[3]               // 11th grade average

. scalar Ybar12 = propatr[4]               // 12th grade average

. gen varatr = ((nstgrade - $ss) / (nstgrade - 1)) * (sdatr^2 / $ss)
```

```
. gen grade_sem = sqrt(varatr)

. scalar Ybar9_sem = grade_sem[1]         // 9th grade sem

. scalar Ybar10_sem = grade_sem[2]        // 10th grade sem

. scalar Ybar11_sem = grade_sem[3]        // 11th grade sem

. scalar Ybar12_sem = grade_sem[4]        // 12th grade sem

. gen weight = nstgrade / stupop          // (N_h / N)

. gen pweight= 1/weight

. gen wpropatr = weight * propatr         // weight strata proportions

. gen wvaratr = weight^2 * varatr         // weight strata variances

. collapse (sum) wpropatr wvaratr         // sum within stata means and vars

. scalar Ybar = wpropatr[1]               // store estimate of pop. at risk

. scalar Ybar_sem = sqrt(wvaratr[1])      // compute root of above measure

. restore

. scalar list Ybar Ybar_sem
      Ybar =  .34547178
  Ybar_sem =  .03225598

. mean atrisk

Mean estimation                   Number of obs   =        200

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
      atrisk |       .345    .033698      .2785491    .4114509
--------------------------------------------------------------

. scalar list Ybar9 Ybar9_sem Ybar10 Ybar10_sem Ybar11 Ybar11_sem Ybar12 Ybar12_sem
     Ybar9 =        .38
 Ybar9_sem =  .06608301
    Ybar10 =  .31999999
Ybar10_sem =  .06322689
    Ybar11 =  .31999999
```

```
Ybar11_sem =  .06330363
    Ybar12 =  .36000001
Ybar12_sem =   .0649146

. mean atrisk, over(grade)

Mean estimation                   Number of obs   =        200

            9: grade = 9
           10: grade = 10
           11: grade = 11
           12: grade = 12

--------------------------------------------------------------
        Over |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
atrisk       |
           9 |        .38   .0693409      .2432627    .5167373
          10 |        .32   .0666395      .1885899    .4514101
          11 |        .32   .0666395      .1885899    .4514101
          12 |        .36   .0685714      .2247801    .4952199
--------------------------------------------------------------

. gen weight = nstgrade / stupop

. gen pweight= 1/weight

. mean atrisk [pweight=pweight]

Mean estimation                   Number of obs   =        200

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
      atrisk |   .3445775   .0337125      .2780979     .411057
--------------------------------------------------------------

. scalar li Ybar
      Ybar =  .34547178
```

Stratified cluster sampling with probability proportional to size

Description

Cluster sampling involves taking a sample where a group of clusters (each of which contains multiple units) is designated, then a random sample of these clusters are drawn. Within each cluster, all units may be included or, in large-scale surveys, a second sample may be drawn. In either case, this last unit is typically the unit of analysis. For example, if we decided to conduct a study by taking a random sample of classrooms within a school, then taking a sample of students within those classrooms, this would be a cluster sampling design. In this example, the classrooms would be the primary sampling unit (*psu*) and the students would be the secondary sampling unit (*ssu*).

Formulas

population size

$$M = \sum_{h=1}^{L} \sum_{i=1}^{N_h} M_{hi}$$

population total

$$Y = \sum_{h=1}^{L} \sum_{i=1}^{N_h} \sum_{j=1}^{M_{hi}} Y_{hij}$$

sample total

$$m = \sum_{h=1}^{L} \sum_{i=1}^{n_h} m_{hi}$$

estimated population total

$$\hat{Y} = \sum_{h=1}^{L} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij} Y_{hij}$$

where

$$w_{hij} = \frac{N_h}{n_h}$$

estimated population size

$$\hat{M} = \sum_{h=1}^{L} \sum_{i=1}^{n_h} \sum_{j=1}^{m_{hi}} w_{hij}$$

estimated population total variance

$$\hat{V}(\hat{Y}) = \sum_{h=1}^{L} (1 - f_h) \frac{n_h}{n_h - 1} \sum_{i=1}^{n_h} (y_{hi} - \bar{y}_h)^2$$

where

$$y_{hi} = \sum_{j=1}^{M_h i} w_{hij} y_{hij}$$

$$\bar{y}_h = \frac{1}{n_h} \sum_{i=1}^{n_h} y_{hi}$$

and

$$f_h = \frac{n_h}{N_h}$$

Example

Using the fake highschool data, let's try to get an estimate of test scores. First, let's take a look at the population values (which, again, we normally don't know):

Estimated means

This time, rather than simply sampling students within each grade, let's sample entire classes. These will be our *PSUs* with students being the *SSUs*. After taking only 10 classes in each grade, we'll compute the mean score within each grade and overall, taking into account the survey design.

```
. insheet using ${datadir}fakehs.csv, clear
(8 vars, 2,003 obs)

. mean testscore                              // overall

Mean estimation                    Number of obs   =      2,003

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
   testscore |   509.0789   1.210999      506.7039    511.4538
```

```
--------------------------------------------------------------

. mean testscore, over(grade)             // within grade

Mean estimation                   Number of obs   =      2,003

            9: grade = 9
           10: grade = 10
           11: grade = 11
           12: grade = 12

--------------------------------------------------------------
        Over |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
testscore    |
           9 |   480.3551   2.132872      476.1723     484.538
          10 |   503.9045    2.26261      499.4672    508.3418
          11 |   515.5706   2.193136      511.2695    519.8716
          12 |   540.0465   2.318895      535.4988    544.5942
--------------------------------------------------------------

. global cut = 10                         // number of classes to keep in each grade

. keep if classid <= $cut                 // keep only sampled classes
(1,131 observations deleted)

. scalar m = _N                           // number of students in sample

. gen weight = nclgrade / $cut            // 1 / (n_h / N_h) or just (N_h / n_h)

. qui sum weight                          // quietly -summarize-

. scalar Mhat = r(sum)                    // store sum of weights

. di Mhat                                 // estimated population
1963

. preserve

. gen wscore = testscore * weight         // (w_hij * y_hij)

. collapse (sum) wscore (first) nstgrade nclgrade, by(grade)

. gen Ybar_grade = wscore / nstgrade      // Ytotal_h / stupop_h = Ybar_h

. scalar Ybar9 = Ybar_grade[1]            // 9th grade average score
```

```
. scalar Ybar10 = Ybar_grade[2]          // 10th grade average score

. scalar Ybar11 = Ybar_grade[3]          // 11th grade average score

. scalar Ybar12 = Ybar_grade[4]          // 12th grade average score

. qui sum wscore                         // quietly -summarize-

. scalar Ybar_school = r(sum) / Mhat     // Ytotal / stupop = Ybar

. restore

. scalar list Ybar9 Ybar10 Ybar11 Ybar12 Ybar_school
     Ybar9 =  475.31216
    Ybar10 =  489.36139
    Ybar11 =  521.64276
    Ybar12 =  515.93848
Ybar_school =  510.17983
```

Estimated standard error of the mean

We can see that our estimate of the population mean, $\hat{M}$, is close to the true value, but not exact. Our within grade estimates aren't that great overall, but the schoolwide estimate is pretty close.

First let's look at an unweighted estimate of the schoolwide sample mean and its standard error.

```
. mean testscore

Mean estimation                   Number of obs   =        872

--------------------------------------------------------------
             |       Mean   Std. Err.     [95% Conf. Interval]
-------------+------------------------------------------------
   testscore |   511.1055   1.852679      507.4693    514.7417
--------------------------------------------------------------
```

Next, let's try compute an estimate of the variance. Note that the equations above speak to the variance of the total. We don't want that. We want the variance of the mean score. Here's what we will do: compute the variance of the total, divide it by the square of the estimated number of *SSUs* to standardize it, and then divide it again by the number of sampled *SSUs* to get the standard error of the estimated mean score. This isn't quite right, as we don't really take into account the clustering of students when we divide, but it will get us a reasonable approximation without recourse to more complicated methods.

```
. gen wscore = testscore * weight         // (w_hij * y_hij)

. gen fpc_r = $cut / nclgrade             // fpc rate by strata (grade)

. collapse (sum) wscore (first) nstgrade fpc_r, by(grade classid) // sum w/n class

. preserve

. collapse (mean) stmscore = wscore, by(grade) // mean weighted score w/n grades

. tempfile stratmeans                          // init temporary file

. save `stratmeans'                            // save temporary file
file /var/folders/h_/wc0n_t2j437g61hxg5t3sbdw1bjh2n/T//S_20497.000012 saved

. restore

. merge m:1 grade using `stratmeans', nogen    // merge grade means into file

    Result                           # of obs.
    -----------------------------------------
    not matched                             0
    matched                                40
    -----------------------------------------

. gen adjsqdiff = (1 - fpc_r) * ($cut / ($cut - 1)) * (wscore - stmscore)^2

. collapse (sum) adjsqdiff                // double sum: w/n strata, overall

. scalar Ybar_school_sem = sqrt(adjsqdiff / Mhat^2) / sqrt(m)

. scalar list Ybar_school Ybar_school_sem
Ybar_school =  510.17983
Ybar_school_sem =  .33002674
```

While our estimate of the schoolwide mean is closer to the true mean than the naive estimation, our standard error is much improved. Great! We should be cautious, however, of the standard error. Wait, what? This is due to the fact that the standard error of complex survey designs cannot be directly computed, only estimated. Our estimate might be too generous. It likely is. There are better, albeit more complicated ways to compute the estimate we want.

Good news

Now that we've gone through this process, the good news is that with most national educational surveys, you won't have to compute weights or figure out means and variances by hand. Instead, the data files will give you the weights you need. Stata also has prepackaged routines to help you in this process. The most important one is `svyset` and its suite of commands. We will discuss these in the next lecture.

```
. log close
      name:  <unnamed>
       log:  /Users/doylewr/lpo_prac/lessons/s1-06-sampling/sampling_part1.log
  log type:  text
 closed on:  30 Sep 2020, 10:28:23
----------------------------------------------------------------------------------

. exit
```