SURV625 Applied Sampling

2025-04-17

SM 625: Week 4 Sampling Project Notes

For each of the three variables that will be the focus of the final course project, the Department of Education would like to generate estimates of means and proportions having a coefficient of variation of no more than 0.05. Using the numbers provided to you in the description of the final project, compute estimates of the element variances for each variable. Given these estimates, compute the desired level of precision (the desired sampling variance) for each estimate that corresponds to the desired coefficient of variation.

Now, given the desired levels of precision for each estimate, compute estimates of the necessary sample sizes for each of these three estimates (assuming simple random sampling), ignoring the finite population correction. These will be starting points for the eventual two-stage cluster sample design.

We first build a table to store our results for each week's assignments.

- We also add the expected averages for each outcome variable.

```
# build dataframe with inputs
MI_school_samples <- tibble(
  Outcome = c("smoked_cig", "smoked_mj", "age_approached_to_smoke"),
  type = c("prop", "prop", "mean"),
  desire_cv = rep(.05, 3),
  expect_mean = c(.25, .15, 12),
)
  # calculate element
MI_school_samples |> kable()
```

Outcome	type	desire_cv	expect_mean
smoked_cig	prop	0.05	0.25
smoked_mj	prop	0.05	0.15
age_approached_to_smoke	mean	0.05	12.00

Our process is to:

- 1st, calculate the estimated element variance.
  - For a proportion, to get the element variance we use $\hat{p}(1-\hat{p})$.
  - For a mean, to get the element variance we simply just square the estimated standard deviation $v(\bar{y}) = \sigma^2$.
- 2nd, we calculate the estimated standard error as $se(\hat{p}) = CV \times \hat{p}$.
- 3rd, we compute the desired sampling variance as: $var(\hat{p}) = se(\hat{p})^2$, where $se(\hat{p}) = \sqrt{var(\hat{p})}$

```
MI_school_samples <- MI_school_samples |>
  mutate(
    # compute element variance
    var = if_else(type=="prop", # for proportions
                                expect_mean * (1 - expect_mean),
                                if_else(type=="mean",# for means
                                        1^2, NA)),
    # compute stand dev
    sd = sqrt(var),
    # compute standard error
    se = desire_cv * expect_mean,
    # compute desired sample variance
    V = se^2
    )

MI_school_samples |> select(-type) |> kable()
```

Outcome	desire_cv	expect_mean	var	sd	se	V
smoked_cig	0.05	0.25	0.1875	0.4330127	0.0125	0.0001563

Outcome	desire_cv	expect_mean	var	sd	se	V
smoked_mj	0.05	0.15	0.1275	0.3570714	0.0075	0.0000562
age_approached_to_smoke	0.05	12.00	1.0000	1.0000000	0.6000	0.3600000

We now estimate the desired sample sizes when we desire a CV =.05 as $n = \frac{s^2}{se^2}$

```
MI_school_samples <- MI_school_samples |>
  mutate(SRS_n = var / V)

MI_school_samples |> select(1, SRS_n) |> kable()
```

Outcome	SRS_n
smoked_cig	1200.000000
smoked_mj	2266.666667
age_approached_to_smoke	2.777778

SM 625: Week 5 Sampling Project Notes

For this week, we will consider the information available for stratified sampling of students. Eventually you are going to design a stratified cluster sample of students, where the clusters (or PSUs) are schools, but we aren't there yet.

Recall the regions of interest in the sampling project description:

```
school_frame <- read_xls(
  "~/work/d/SURV625project/data/MI_school_frame_head_counts.xls")
```

Region	County_ID
1	07, 31, 66
2	22, 27, 36, 55
3	02, 21, 52
4	17, 48, 49, 77
5	01, 04, 06, 16, 20, 26, 35, 60, 65, 68, 69, 71, 72
6	05, 10, 15, 18, 24, 28, 40, 43, 45, 51, 53, 57, 67, 83
7	03, 08, 11, 12, 13, 14, 34, 39, 41, 54, 59, 61, 62, 64, 70, 75, 80
8	09, 19, 23, 25, 29, 30, 33, 37, 38, 46, 47, 56, 73, 78, 81
9	32, 44, 50, 58, 63, 74, 76, 79, 82

As "State officials are interested in providing, if at all possible, separate estimates for each of nine education regions in the state, where the regions are defined by groups of counties", we will use these nine regions as strata.

Prepare a table that includes the:

- Overall population counts in each of these nine strata (the total count of students in the target population at each school is in the tot_all column on the sampling frame).
- Given these counts, once you have the working overall sample size (unknown for now and will be decided by your team next week), what is the proportionate allocation plan of that sample of students across these nine strata?

```
# we will use Region, County_ID, and tot_all

# region counts
strata_Prop_allocate <- school_frame |>
  group_by(Region) |>
```

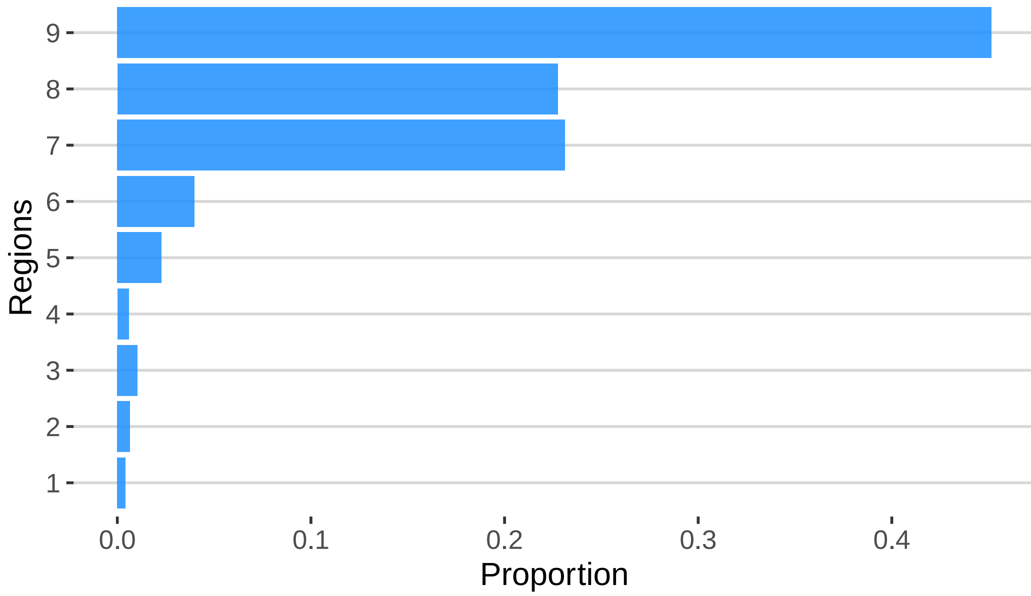
```r
  reframe(M_h = sum(tot_all), # total of students in stratum
          N_h = n()) |> # total of schools in stratum
  mutate(prop_allocation = M_h/sum(M_h))

strata_Prop_allocate |>
  kable()
```

Region	M_h	N_h	prop_allocation
1	3561	20	0.0042896
2	5474	30	0.0065941
3	8631	33	0.0103971
4	4855	31	0.0058484
5	18907	80	0.0227757
6	33133	133	0.0399126
7	191992	644	0.2312772
8	188830	549	0.2274682
9	374755	923	0.4514370

```r
# what is the proportionate allocation plan of that sample
## of students across these nine strata?
```

```r
strata_Prop_allocate |>
  mutate(Region = factor(Region)) |>
  ggplot(aes(x=Region, y=prop_allocation)) +
  geom_col(position="dodge", fill="dodgerblue", alpha=.85) +
  coord_flip() +
  guides(fill=guide_legend(title="", reverse = TRUE)) +
  labs(
    title = "Figure 1. Proportionate Allocation Plan Across Nine Strata",
    x = "Regions",
    y = "Proportion"
  ) +
  theme_hc()
```

Figure 1. Proportionate Allocation Plan Across Nine Strata



SM 625: Week 6 Sampling Project Notes

From a previous study, you obtain estimates of the following design effects for each of these three estimates:

- proportion ever smoked one cigarette = 2.5;
- proportion ever smoked marijuana = 2.0; and
- mean age when first asked to smoke = 1.7.

This previous study featured a sample of size n = 7,500 students between the ages of 13 and 19, selected from a total of a = 150 clusters. Using this information, compute a synthetic estimate of roh for each of the three variables. These synthetic estimates of roh will be used to consider alternative cluster sample designs as you continue with your project work. Finally, budget and cost information is now available. The total budget for data collection for this project will be $500,000. The client and the data collection organization estimate that the data collection will cost $3,000 per primary stage cluster (school), and $50 per completed questionnaire within a cluster. We will use this cost information moving forward for optimal subsample size calculations.

We can estimate the sample ICC or roh from the given design effect estimate as:

$$\hat{roh} = \frac{deff - 1}{m - 1}$$

We now that the sample total is $nm = 7500$ and the sample number of cluster is $n = 150$, which we can take the mean cluster size as $m = nm/n = 7500/150 = 50$ and use it to calculate roh.

```r
nm <- 7500
n <- 150
m <- nm / n

MI_school_samples <- MI_school_samples |>
  # add deff and roh to our table
  mutate(desire_deff = c(2.5, 2.0, 1.7),
         # compute roh
         roh = (desire_deff - 1) / (m - 1),
```

```r
        roh = round(roh, 4)
        )

MI_school_samples |> select(Outcome, desire_deff, roh) |>  kable()
```

Outcome	desire_deff	roh
smoked_cig	2.5	0.0306
smoked_mj	2.0	0.0204
age_approached_to_smoke	1.7	0.0143

SM 625: Week 7 Sampling Project Notes

Recall that the client and the data collection organization estimated that the data collection would cost $3,000 per primary stage cluster (school), and $50 per completed questionnaire within a cluster. We will now use this information for optimum subsample size calculations. Recall that the total budget for data collection will be $500,000.

Given this cost information and your estimates of roh for the three different variables of primary interest from last week, compute the optimum subsample size (and the corresponding optimal number of first stage clusters, given the total budget above) for each of the variables.

- We now have budget constraints and denote the cost per cluster as $c_n = \$3,000$ and cost per element as $c_m = \$50$, with a total budget constraint of $C = \$500,000$. Since we know there are $n = 150$ clusters and a total sample size of $7,500$ students.
- To compute the optimum m size we use the following equation:

$$m_{opt} = \sqrt{\frac{c_n}{c_m}\frac{1-roh}{roh}}$$

```
c_n = 3000 # cost per cluster
c_m = 50 # cost per element within cluster
C = 500000 # total budget

MI_school_samples <- MI_school_samples |>
  mutate(
    # compute optimum m size
    m_opt = sqrt( (c_n / c_m) * ( (1-roh)/roh) ),
    n_opt = C / (c_n + m_opt * c_m),
    # compute new deff
    deff_new = 1 + (m_opt-1) * roh,
     # compute total SSU
    total_nm = m_opt * n_opt)

MI_school_samples |> select(Outcome, m_opt, n_opt) |>
  kable()
```

Outcome	m_opt	n_opt
smoked_cig	43.59799	96.52697
smoked_mj	53.67659	87.96886
age_approached_to_smoke	64.31022	80.44391

How will you decide on a single overall optimum subsample size to use in your design?

- Above we estimated the new design effects which range from 2.3 to 1.9, which are almost in line with our desired design effects of 2.5, 1.7. Below we print the new design effects, optimum number of cluster and cluster size, total sample size `total_nm`for our projected $500,000 budget for all three outcome variables.
  - Finally, we compute the sampling cost as $n \times c_n + n \times m \times c_m$ which we defined these terms above.

```
MI_school_samples |>
  select(Outcome, m_opt, n_opt, deff_new, total_nm) |>
  mutate_at(c(2, 3, 5), floor) |>
  mutate(cost = (c_n * n_opt) + (c_m * n_opt * m_opt),
         cost = scales::dollar(cost),
         Outcome = ifelse(Outcome == "age_approached_to_smoke",
                          "age_smoke", Outcome),
           deff_new = round(deff_new, 4)) |>
  kable()
```

Outcome	m_opt	n_opt	deff_new	total_nm	cost
smoked_cig	43	96	2.3035	4208	$494,400
smoked_mj	53	87	2.0746	4721	$491,550
age_smoke	64	80	1.9053	5173	$496,000

Think about a comparison of alternative cluster sample designs: under a fixed cost constraint, how would we decide which design would be best? What will be your overall sample size (n) under this new optimum subsample size?

As you make progress in writing up what you have done so far, provide some discussion of the rationale for your choices in this regard.

Next, given this optimum subsample size and treating the values of roh as portable, compute the new expected DEFF for each estimate given the new design (this can be specific to each variable / estimate, given the different optimum subsample sizes). In addition, compute a new expected SRS variance for each variable under the new design, using the new "optimum" overall sample size (remember that you can treat the element variances for each variable estimated last week as portable). Finally, compute the new expected sampling variance for each estimate under this new cluster sample design. Are you still meeting the client's precision requirements?

- Given that we have three outcome variables, we also have three optimum number of clusters and cluster size estimates. That is, we can design and examine three options of different optimum number of clusters and cluster sizes.
- We will use the portable roh estimate and calculate new design effects, SRS variance, and complex design variance for each outcome variable.

```
map(seq(1,3), function(x){

  MI_school_samples |>
  select(Outcome, roh, m_opt, n_opt, total_nm) |>
  # we can print projected total cost for n=50
  mutate(m_opt = m_opt[x], # optimum m from first row
         n_opt = n_opt[x],
         total_nm = m_opt*n_opt,
         # calculate new deff
         deff_new = 1 + (m_opt-1) * roh,
         # recalcualte element variance
         var = c(.24, .1275, 9),
         # calcualte SRS variance
         var_srs =  var / (total_nm - 1),
         # calculate complex design variance
         var_crs = var_srs * deff_new,
         #m_opt = round(m_opt)
         ) |>
    select(-roh, -var)  |>
    mutate_at(2:3, floor) |>
    mutate(total_nm = m_opt*n_opt) |>
    mutate_at(5:7, round, 5) |>
  kable()

}) |>
  set_names(str_c(rep("Option ", 3), seq(1,3)))
```

```
$`Option 1`


|Outcome                  | m_opt| n_opt| total_nm| deff_new| var_srs| var_crs|
|:------------------------|-----:|-----:|--------:|--------:|-------:|-------:|
|smoked_cig               |    43|    96|     4128|  2.30350| 0.00006| 0.00013|
|smoked_mj                |    43|    96|     4128|  1.86900| 0.00003| 0.00006|
|age_approached_to_smoke  |    43|    96|     4128|  1.60915| 0.00214| 0.00344|
```

```
$`Option 2`


|Outcome                  | m_opt| n_opt| total_nm| deff_new| var_srs| var_crs|
|:------------------------|-----:|-----:|--------:|--------:|-------:|-------:|
|smoked_cig               |    53|    87|     4611|  2.61190| 0.00005| 0.00013|
|smoked_mj                |    53|    87|     4611|  2.07460| 0.00003| 0.00006|
|age_approached_to_smoke  |    53|    87|     4611|  1.75328| 0.00191| 0.00334|

$`Option 3`


|Outcome                  | m_opt| n_opt| total_nm| deff_new| var_srs| var_crs|
|:------------------------|-----:|-----:|--------:|--------:|-------:|-------:|
|smoked_cig               |    64|    80|     5120|  2.93729| 0.00005| 0.00014|
|smoked_mj                |    64|    80|     5120|  2.29153| 0.00002| 0.00006|
|age_approached_to_smoke  |    64|    80|     5120|  1.90534| 0.00174| 0.00332|
```

We print standard error for the complex design with 95% confidence intervals, and we also flag whether the sampling variance from the clustering is equal or smaller than the desired sampling variance.

```r
map(seq(1,3), function(x){

  MI_school_samples |>
  select(Outcome, expect_mean, V, roh, m_opt, n_opt, total_nm) |>
  # we can print projected total cost for n=50
  mutate(m_opt = m_opt[x], # optimum m from first row
         n_opt = n_opt[x],
         total_nm = m_opt*n_opt,
         # calculate new deff
         deff_new = 1 + (m_opt-1) * roh,
        # recalcualte element variance
         var = c(.24, .1275, 9),
         # calcualte SRS variance
         var_srs =  var / (total_nm - 1) ,
         # calculate complex design variance
         var_crs = var_srs * deff_new,
         # compute confidence intervals
         se = sqrt(var_crs),
         lower = expect_mean  - 1.96*se,
```

```r
         upper = expect_mean + 1.96*se,
        # flag if var_crs is lower or = to desired sampling var
         var_ck = ifelse(var_crs <= V, "yes", "no")) |>
  select(Outcome, expect_mean, se, lower, upper, var_ck) |>
  mutate_at(3:5, round, 4) |>
  kable()

}) |>
  set_names(str_c(rep("Option ", 3), seq(1,3)))
```

```
$`Option 1`


|Outcome                  | expect_mean|     se|   lower|   upper|var_ck |
|:------------------------|-----------:|------:|-------:|-------:|:------|
|smoked_cig               |        0.25| 0.0115|  0.2275|  0.2725|yes    |
|smoked_mj                |        0.15| 0.0075|  0.1352|  0.1648|no     |
|age_approached_to_smoke  |       12.00| 0.0587| 11.8850| 12.1150|yes    |

$`Option 2`


|Outcome                  | expect_mean|     se|   lower|   upper|var_ck |
|:------------------------|-----------:|------:|-------:|-------:|:------|
|smoked_cig               |        0.25| 0.0115|  0.2274|  0.2726|yes    |
|smoked_mj                |        0.15| 0.0075|  0.1353|  0.1647|yes    |
|age_approached_to_smoke  |       12.00| 0.0578| 11.8867| 12.1133|yes    |

$`Option 3`


|Outcome                  | expect_mean|     se|   lower|   upper|var_ck |
|:------------------------|-----------:|------:|-------:|-------:|:------|
|smoked_cig               |        0.25| 0.0117|  0.2271|  0.2729|yes    |
|smoked_mj                |        0.15| 0.0075|  0.1353|  0.1647|no     |
|age_approached_to_smoke  |       12.00| 0.0576| 11.8871| 12.1129|yes    |
```

- Option 2 with a number of cluster of 87 and cluster size of 53 is the design we will choose given that the total sample size of 4,611 is within the allocated budget ($491,550).

We prefer this model because it stays close to the desired design effects we received from the customer. Additionally, the standard errors we estimate for this second option overall are the smallest resulting in tighter 95% confidence intervals for the expected estimates we were provided. This design in close to option 3, yet we prefer having a slightly smaller SSU if we can increase the number of PSUs sampled since this gives us a cost efficiency.

The client has also provided other new information: the estimated size of the target population is N = 830,138. Given this population size and your overall sample size (n) under the new optimum subsample size computed above, what is your overall working sampling fraction (f)? Does it seem like finite population corrections will be necessary in your sampling variances if you choose to perform SRSWOR at some point?

```
# total pop
N <- 830138

# optimum n
total_nm <- MI_school_samples |>
  slice(2) |> # second design option
  summarise(m_opt*n_opt) |>
  pull()


samp_frac <- total_nm / N; samp_frac
```

```
[1] 0.005688052
```

```
MI_school_samples_table <- MI_school_samples |>  select(Outcome, expect_mean, roh,
                             m_opt, n_opt, total_nm) |>
  # we can print projected total cost for n=50
  mutate(m_opt = m_opt[2], # optimum m from first row
         n_opt = n_opt[2],
         total_nm = m_opt*n_opt,
         # calculate new deff
         deff_new = 1 + (m_opt-1) * roh,
         # recalcuualte element variance
         var = c(.24, .1275, 9),
         # calcualte SRS variance with sampl_fraction
         var_srs =(1 - samp_frac) * var / (total_nm - 1),
         # calculate complex design variance
         var_crs = var_srs * deff_new,
```

```
         # compute confidence intervals
         se = sqrt(var_crs),
         lower = expect_mean  - 1.96*se,
         upper = expect_mean + 1.96*se )

MI_school_samples_table |>
  select(Outcome, var_crs, se, lower, upper) |>
  mutate_at(2:4, round, 5) |>
  kable()
```

Outcome	var_crs	se	lower	upper
smoked_cig	0.00013	0.01149	0.22748	0.2725212
smoked_mj	0.00006	0.00746	0.13537	0.1646295
age_approached_to_smoke	0.00332	0.05765	11.88701	12.1129933

Our overall sampling fraction is .0057. In examining the complex design variances, and recalculating the expected standard error and 95% confidence interval given the sampling raction, it does not appear that accounting for a population correction makes a huge impact, and we suggest it will not be necessary in our sampling variance for an SRSWOR design

SM 625: Week 8 Sampling Project Notes

Assume that you will decide to allocate your final computed n_{opt} number of clusters to each of the nine project strata based on the proportions of the total number of students in the population in each stratum (i.e., if 20% of the population of students comes from Region 1, you would sample 20% of your clusters from that region). Describe the first-stage sampling fractions for each stratum, where the total number of schools to sample at the first stage in each stratum is defined by your proportionate allocation of the n_{opt} clusters.

Next your team should extend your design to consider stratified PPeS selection of schools from each of the nine strata at the first stage of your sample design.

You have been provided with a sampling frame that lists the schools within each region. Given the information on the sampling frame, how might you sort this list to achieve implicit stratification within the regions? You can treat the overall student count from a previous year (tot_all) as the measure of size for the PPeS sampling. Given this information, compute your zone size for systematic PPeS sampling within each of the nine strata (regions), and proceed with systematic selection based on fractional intervals to select the allocated number of schools within each stratum using PPeS sampling. What is your first-stage sampling fraction within each of the nine strata?

- Using the proportionate allocation by strata computed earlier, we assign and add cluster allocation by stratum by $n_{opt} \times prop - allocation$.
- nonresponse adjustment is achieved by taking our optimum values and adjusting them by the amount of respondents that are likely to complete the survey.
- We also calculate the zone size which we label as k_h as:

$$k_h = \frac{nMOS_i}{\sum_t MOS_i}$$

```
set.seed(9999)

# response rates
school_rr <- .30
student_rr <- .70
# Given values
n_opt <- MI_school_samples |>
  slice(2) |> select(n_opt) |>
  pull() / school_rr

m_opt <- MI_school_samples |>
  slice(2) |> select(m_opt) |>
  pull() / student_rr
```

```
# Compute proportional allocation of clusters to each stratum
region_summary <- strata_Prop_allocate |>
  # Ensure at least 1 cluster per
  mutate(n_h = round(n_opt * prop_allocation)#,
           # we need to adjust the last n_h to get an exact 290
         #n_h = ifelse(n_h == 131, n_h-1, n_h)
         ) |>
  mutate(N_h = as.double(N_h)) |>
  group_by(Region) |>
  reframe(across(where(is.double), ~ sum(.x))) |>
  mutate(
         f_h = n_h / N_h,     # sampling fraction
         k_h =  round(M_h / n_h)) |>    # zone size
  # create random start values
  rowwise() |>
  mutate(RN = sample(1:k_h, 1)) |>
  ungroup() |>
  mutate_at(vars(prop_allocation, f_h), round, 3)


region_summary |>
  select(Region, prop_allocation, n_h, k_h, RN) |>
  kable()
```

Region	prop_allocation	n_h	k_h	RN
1	0.004	1	3561	3168
2	0.007	2	2737	2310
3	0.010	3	2877	1321
4	0.006	2	2428	131
5	0.023	7	2701	2122
6	0.040	12	2761	2114
7	0.231	68	2823	374
8	0.227	67	2818	380
9	0.451	132	2839	1673

- To achieve implicit stratification we order the school list sorted by size of student in each region. To compute zone size we use

```
# sort list of schools by student size
school_frame_sorted <- school_frame |>
  mutate(hs = sum(g9_totl, g10_totl, g11_totl, g12_totl) / tot_all) |>
  arrange(desc(hs)) |>
  select(-hs)

min_MOS <- m_opt

# create vectors of selection values for each stratum
RN_sample <- map(1:nrow(region_summary), function(x){

  # pass table created in last code chunk
  round(seq(region_summary$RN[x], # random start
      region_summary$M_h[x], # total number of students
      region_summary$k_h[x])) # k sampling interval
})
```

```
# we link the selected blocks
dat <- school_frame_sorted |>
  group_by(Region) |>
  mutate(
    # assing ids
    id = row_number(),
    # flag if minimum MOS not met
    min_m_req = ifelse(tot_all >= min_MOS, 1 , 0),
    # create links and convert to clusters
    linking = lead(min_m_req, default=1),
    # assign clustering
    cluster = cumsum(lag(linking, default=1)),
    # add cumulative counts
    cumulative_max = cumsum(tot_all),
    cumulative_min = 1 + lag(cumulative_max, default = 0) )
```

```
# for each region loop through RN_sample & assign selection to schools
dat_selected <- map_dfr(1:9, function(x){

  dat |>
    filter(Region %in% x) |>
    add_column(RN_sample[[x]] |> tibble() |> data.table::transpose()) |>
    # create flag for blocks that are selected
    mutate(selected =
             as.numeric(if_any(starts_with("V"), ~
```

```r
                        between(.x, cumulative_min, cumulative_max)))) |>
    # drop select population elements
    select(-starts_with("V"))

})
```

```r
# this is where schools are linked
dat_linked <- dat_selected |>
  group_by(Region) |>
  mutate(
    # flag if minimum MOS not met
    min_n_req = ifelse(tot_all >= min_MOS, 1 , 0),
    # create links and convert to clusters
    linking = lead(min_n_req, default=1),
    # assign clustering
    cluster = cumsum(lag(linking, default=1))) |>
  ungroup()
```

```r
# show cluster of blocks selected, total HUs
sample_selected <- map_dfr(1:9, function(x){

  linkage = dat_linked |>
    filter(Region %in% x, selected == 1) |>
    select(Region, cluster) |>
    mutate(Selection = RN_sample[[x]]) |>
    pull(cluster)

  dat = dat_linked |>
    filter(Region %in% x,
           cluster %in% linkage) |>
    mutate(MOS = as.numeric(tot_all)) |>
    group_by(cluster) |>
    mutate(
      cluster = cur_group_id(),
      total_MOS = sum(MOS, na.rm = TRUE)
    ) |>
    arrange(desc(id))  # optional: sort within cluster

  if (x == 1) {
    # get unique cluster id from Region 1
    first_cluster_id <- dat |>
      filter(Region == 1) |>
```

```r
      pull(cluster) |>
      unique() |>
      min()

    # filter the first cluster
    first_cluster <- dat |> filter(cluster == first_cluster_id)

    # split it into two halves (or roughly)
    n <- nrow(first_cluster)
    first_half <- first_cluster[1:floor(n/2), ] |>
      mutate(
        SECU = "1A",
        SECU_MOS = sum(MOS/2, na.rm = TRUE)
      )
    second_half <- first_cluster[(floor(n/2) + 1):n, ] |>
      mutate(
        SECU = "1B",
        SECU_MOS = sum(MOS/2, na.rm = TRUE)
      )

    # everything else from Region 1
    remaining <- dat |> filter(cluster != first_cluster_id) |>
      mutate(SECU = as.character(cluster),
             SECU_MOS = total_MOS)

    # combine all Region 1 units
    dat <- bind_rows(first_half, second_half, remaining)
  } else {
    dat <- dat |>
      mutate(
        SECU = as.character(cluster),
        SECU_MOS = total_MOS
      )
  }

  return(dat)
}) |>
  ungroup()

sample_selected <- sample_selected |>
  slice(-1)
```

```r
# save data to github
write_xlsx(sample_selected,
           "~/work/d/SURV625project/data/sample_selected.xlsx")
```

```r
# Form Pseudo-Strata for Paired Selection Model
pseudo_strata_df <- sample_selected |>
  group_by(Region) |>
  arrange(desc(MOS)) |>   # optionally sort by size for pairing
  mutate(row_in_group = row_number(),
         pseudo_stratum_id = paste0("R", Region, "_P", ceiling(row_in_group / 2))) |>
  ungroup()


# how many pseudo strata in each region?
pseudo_strata_df |>
  group_by(Region) |>
  distinct(pseudo_stratum_id) |>
  count() |>
  ungroup() |>
  mutate(Pseudo_strata = c(1, 1, 1, 2, 2, 2, 2, 3, 4)) |>
  group_by(Pseudo_strata) |>
  reframe(Secu = sum(n)/2) |>
  kable()
```

Pseudo_strata	Secu
1	2
2	79
3	56
4	95

SM 625: Week 10 Sampling Project Notes

There are four primary tasks for your team to consider over the next week:

1. Given your overall m_{opt} n_opt and N (based on the sampling frame), you've already computed the overall sampling fraction, . For each of the nine strata, compute the required number of students to subsample from each sampled school based on the stratified PPeS design in order to maintain epsem across all strata.

- Within strata, retain epsem for stratified PPS sampling across strata $f = f_h$ for all h.

$$f_h = \frac{n_h MOS_{hi}}{\sum_{,i \in h} MOShi} \frac{m_h^*}{MOS_{hi}}$$

```
# Required Students per School (m_h_star) to Maintain EPSEM:
region_summary<- region_summary |>
  mutate(m_h_star=c(samp_frac*k_h))

region_summary |>
  select(Region, k_h, RN, m_h_star) |>
  kable()
```

Region	k_h	RN	m_h_star
1	3561	3168	20.25515
2	2737	2310	15.56820
3	2877	1321	16.36453
4	2428	131	13.81059
5	2701	2122	15.36343
6	2761	2114	15.70471
7	2823	374	16.05737
8	2818	380	16.02893
9	2839	1673	16.14838

2. Do each of the schools that you sampled in a given region have the minimum sufficient size, given the stratum-specific subsample sizes computed in Task #1? Do subsequent schools on the list have the minimum sufficient size? If not, what will you do?

```
region_min_MOS <- region_summary %>%
  group_by(Region) %>%
  mutate(
    min_MOS2 = ceiling(m_h_star /  0.7)  # Total response rate = 0.21, expanded sample size
```

```
  )

# Processing schools by region and generating clusters of links
linked_schools <- sample_selected %>%
  left_join(region_min_MOS, by = "Region") %>%  # Combined Minimum MOS
  group_by(Region) %>%
  mutate(
    # Initialize cumulative MOS and link tags
    cumulative_mos = cumsum(tot_all),
    need_link = if_else(tot_all < min_MOS2, 1, 0),
    # Dynamic generation of cluster IDs: linking when cumulative MOS is insufficient
    cluster_id = cumsum(
      if_else(
        cumulative_mos - lag(cumulative_mos, default = 0) >= min_MOS2 | row_number() == 1,
        1, 0
      )
    )
  ) %>%
  ungroup()

# how many linked clusters by region
linked_schools |>
  group_by(Region) |>
  count(cluster_id) |>
  count() |>
  ungroup() |>
  kable()
```

Region	n
1	1
2	2
3	3
4	10
5	7
6	12
7	177
8	154
9	270

```r
# Summarize the total MOS for each cluster and check for compliance
cluster_summary <- linked_schools %>%
  group_by(Region, cluster_id) %>%
  summarise(
    total_mos = sum(tot_all),
    schools = toString(BCODE),
    min_MOS2 = first(min_MOS2),
    .groups = "drop"
  ) %>%
  mutate(
    sufficient = if_else(total_mos >= min_MOS2, "Yes", "No")
  )
# Output clusters that need to be relinked (total MOS still insufficient)
clusters_to_relink <- cluster_summary %>% filter(sufficient == "No")

# Recursive linking until all clusters are up to standard
while (nrow(clusters_to_relink) > 0) {
  linked_schools <- linked_schools %>%
    group_by(Region) %>%
    mutate(
      cluster_id = if_else(
        cluster_id %in% clusters_to_relink$cluster_id,
        cluster_id + 1,  # Merge to the next cluster
        cluster_id
      )
    ) %>%
    ungroup()

  # Summary of recomputation clusters
  cluster_summary <- linked_schools %>%
    group_by(Region, cluster_id) %>%
    summarise(
      total_mos = sum(tot_all),
      schools = toString(BCODE),
      min_MOS2 = first(min_MOS2),
      .groups = "drop"
    ) %>%
    mutate(sufficient = if_else(total_mos >= min_MOS2, "Yes", "No"))

  clusters_to_relink <- cluster_summary %>% filter(sufficient == "No")
}
```

```r
final_clusters <- linked_schools %>%
  group_by(Region, cluster_id) %>%
  summarise(
    linked_schools = paste(BCODE, collapse = ", "),
    total_mos = sum(tot_all),
    min_MOS2 = first(min_MOS2),
    .groups = "drop"
  ) %>%
  mutate(
    status = if_else(total_mos >= min_MOS2, "Valid", "Invalid")
  )


linked_schools <- linked_schools %>%
  left_join(
    cluster_summary %>% select(Region, cluster_id, total_mos),
    by = c("Region", "cluster_id")
  )

# Print results
final_clusters |>
  select(Region:total_mos) |>
  kable(col.names = c("h", "id", "n", "m"))
```

h	id	n	m
1	1	00497	430
2	1	02039	675
2	2	02040	258
3	1	01155	928
3	2	01527	427
3	3	04860	197
4	1	02692	323
4	2	06812	64
4	3	08446	57
4	4	08063	49
4	5	03998	47
4	6	04034	46
4	7	09308	43
4	8	02305	41
4	9	08521	33
4	10	07124, 04506, 01509, 07718, 09119	69

h	id	n	m
5	1	00075	1500
5	2	04438	861
5	3	01769	696
5	4	01482	580
5	5	04516	342
5	6	06369	259
5	7	05860	162
6	1	00554	2078
6	2	08470	1859
6	3	03017	1031
6	4	07724	649
6	5	02339	569
6	6	03018	479
6	7	01417	398
6	8	02333	355
6	9	05811	264
6	10	02155	233
6	11	00438	174
6	12	07271	113
7	1	03246	1950
7	2	05974	1849
7	3	01463	1830
7	4	06117	1598
7	5	03095	1412
7	6	02652	1373
7	7	02106	1359
7	8	02587	1326
7	9	02272	1315
7	10	02275	1260
7	11	01498	1212
7	12	02439	1140
7	13	02704	1037
7	14	04404	1007
7	15	00823	1003
7	16	05882	973
7	17	01336	962
7	18	01013	932
7	19	04176	914
7	20	01576	886
7	21	01462	868
7	22	04610	837

h	id	n	m
7	23	00882	831
7	24	04181	788
7	25	01324	774
7	26	00286	748
7	27	03515	735
7	28	00420	702
7	29	06178	688
7	30	00907	663
7	31	00322	654
7	32	00408	636
7	33	08802	612
7	34	06306	592
7	35	03204	566
7	36	01757	550
7	37	03454	532
7	38	04906	509
7	39	00435	500
7	40	02651	484
7	41	01430	477
7	42	02019	468
7	43	00501	454
7	44	00775	442
7	45	03135	430
7	46	01091	424
7	47	08890	404
7	48	01031	389
7	49	04398	372
7	50	08423	360
7	51	02148	342
7	52	02318	330
7	53	00888	322
7	54	02004	313
7	55	00296	292
7	56	02740	271
7	57	06750	258
7	58	06086	239
7	59	03455	218
7	60	00817	201
7	61	08450	177
7	62	04366	161
7	63	09912	142

h	id	n	m
7	64	03994	125
7	65	00620	104
7	66	07290	77
7	67	00576	76
7	68	08372	75
7	69	09913	74
7	70	08967	74
7	71	08817	74
7	72	02193	74
7	73	03546	72
7	74	06322	70
7	75	01731	69
7	76	01448	69
7	77	04967	68
7	78	06741	67
7	79	03812	67
7	80	05487	64
7	81	08948	63
7	82	08161	63
7	83	07764	62
7	84	03681	62
7	85	09922	61
7	86	08576	61
7	87	04032	61
7	88	03624	61
7	89	05790	60
7	90	04787	60
7	91	01516	60
7	92	02585	59
7	93	09898	58
7	94	08919	58
7	95	08530	57
7	96	05839	57
7	97	03362	57
7	98	07389	56
7	99	02366	53
7	100	05794	51
7	101	09562	50
7	102	09309	49
7	103	07256	49
7	104	06730	49

h	id	n	m
7	105	07943	47
7	106	07765	47
7	107	07289	47
7	108	05619	47
7	109	01735	47
7	110	00671	47
7	111	03963	46
7	112	03713	46
7	113	02577	46
7	114	09531	45
7	115	06133	45
7	116	04214	45
7	117	08227	44
7	118	06608	44
7	119	01233	44
7	120	03854	43
7	121	08410	42
7	122	05428	42
7	123	01398	42
7	124	00685	42
7	125	08923	41
7	126	06434	41
7	127	09525	40
7	128	06215	40
7	129	03820	40
7	130	04929	39
7	131	09757	38
7	132	07293	38
7	133	09002	37
7	134	07917	37
7	135	04020	37
7	136	03699	37
7	137	04002	36
7	138	03357	36
7	139	03185	36
7	140	00241	36
7	141	09699	35
7	142	09304	35
7	143	03656	35
7	144	09471	34
7	145	03881	32

h	id	n	m
7	146	09542	31
7	147	05106	31
7	148	03778	31
7	149	00108	31
7	150	03360	30
7	151	09329	29
7	152	09149	29
7	153	07935	29
7	154	05387	29
7	155	03885	29
7	156	01200	29
7	157	09129	28
7	158	07391	28
7	159	05940	28
7	160	02916	28
7	161	08556	27
7	162	07684	26
7	163	06524	26
7	164	05342	26
7	165	02904	26
7	166	02787	26
7	167	08212	25
7	168	08165	25
7	169	05480	25
7	170	05470	25
7	171	09640	24
7	172	09068	24
7	173	08007	24
7	174	08881	23
7	175	04014	23
7	176	03732	23
7	177	01726, 08973, 06031, 03629, 09635, 08239, 07784, 07005, 03680, 09598, 08583, 01823, 09766, 09764, 09107, 08994, 01829, 04209, 00426, 08900, 08442, 03945, 02680, 08367, 07294, 04756, 04208, 09743, 05455, 08539, 08231, 07617, 06934, 04703, 04022, 03850, 01850, 08781, 07404, 04629, 09176, 08666, 08321, 07890, 07841, 06976, 06451, 05841, 03779, 02901, 09903, 09085, 06159, 03959, 00722, 00010, 09410, 08758, 05327, 05297, 03217, 03099, 09394, 06711, 05231, 05169, 05136, 03974, 03526, 09768, 08860, 08837, 01743, 00222, 08236, 06332, 03783, 00102, 09369, 09076, 09075, 04691, 04332, 03311, 06473, 05458, 05041, 04750, 09712, 08179, 07267, 06908, 06048, 06020, 05307, 05187, 04576, 09523, 08955, 08883, 07288, 06602, 06461, 05291, 00602	1075
8	1	02436	2580

h	id	n	m
8	2	04882	2488
8	3	00402	2243
8	4	05671	1946
8	5	05158	1779
8	6	06203	1772
8	7	01457	1725
8	8	05009	1642
8	9	01166	1623
8	10	01256	1537
8	11	00227	1529
8	12	05157	1431
8	13	01044	1407
8	14	01711	1391
8	15	06273	1210
8	16	05690	1201
8	17	01993	1163
8	18	00027	1111
8	19	02426	1082
8	20	05012	1052
8	21	05625	979
8	22	01060	967
8	23	00656	950
8	24	01951	924
8	25	02777	914
8	26	02231	898
8	27	01319	870
8	28	02957	847
8	29	00031	808
8	30	05763	765
8	31	08715	728
8	32	07691	697
8	33	04110	656
8	34	09382	649
8	35	06233	631
8	36	08039	593
8	37	03013	578
8	38	06282	569
8	39	05696	545
8	40	02920	532
8	41	00398	519
8	42	00536	504

h	id	n	m
8	43	02774	495
8	44	08712	481
8	45	05138	475
8	46	05818	460
8	47	03393	447
8	48	00746	435
8	49	02633	419
8	50	00190	399
8	51	04515	385
8	52	02685	361
8	53	00267	345
8	54	08049	318
8	55	00768	304
8	56	02919	280
8	57	06656	263
8	58	09005	244
8	59	00606	221
8	60	07496	201
8	61	01284	181
8	62	08739	158
8	63	02606	133
8	64	06287	113
8	65	07431	91
8	66	04624	76
8	67	02891	76
8	68	03694	75
8	69	08731	74
8	70	08571	69
8	71	04372	67
8	72	01827	67
8	73	09349	66
8	74	08058	66
8	75	04074	66
8	76	09446	65
8	77	08777	62
8	78	07631	62
8	79	07726	61
8	80	03666	60
8	81	09752	56
8	82	07894	55
8	83	01283	55

h	id	n	m
8	84	09897	53
8	85	07973	53
8	86	09833	51
8	87	09711	51
8	88	08826	51
8	89	06326	50
8	90	08721	48
8	91	06606	48
8	92	03759	48
8	93	00068	46
8	94	09317	45
8	95	08653	45
8	96	03920	45
8	97	08716	44
8	98	02900	44
8	99	07801	43
8	100	03764	43
8	101	06889	42
8	102	04981	41
8	103	03577	41
8	104	07412	40
8	105	09779	39
8	106	08314	39
8	107	08241	39
8	108	03989	39
8	109	03611	39
8	110	09923	38
8	111	08857	38
8	112	08323	38
8	113	07585	38
8	114	05808	38
8	115	01816	38
8	116	08570	37
8	117	07743	37
8	118	04717	37
8	119	03953	37
8	120	09083	36
8	121	04956	36
8	122	08659	35
8	123	08322	35
8	124	04029	35

h	id	n	m
8	125	07156	34
8	126	09747	33
8	127	08727	33
8	128	08655	33
8	129	06415	32
8	130	04027	31
8	131	04018	31
8	132	03956	31
8	133	03947	30
8	134	03846	30
8	135	01830	30
8	136	00339	30
8	137	08548	29
8	138	07742	29
8	139	05434	29
8	140	04707	29
8	141	03978	29
8	142	07882	28
8	143	03991	28
8	144	09918	27
8	145	09849	26
8	146	09484	26
8	147	08590	25
8	148	08519	25
8	149	08664	24
8	150	08580	24
8	151	07871	24
8	152	07810	24
8	153	03913	24
8	154	07502, 05482, 07410, 07016, 06734, 05391, 04699, 09374, 06736, 03359, 02121, 00360, 08688, 08216, 04391, 03961, 04960, 03976, 04190, 00012, 08755, 06883, 05984, 05567, 05439, 04261, 01831, 01730, 09668, 09478, 08493, 06220, 03788, 01449, 09725, 06024, 04207, 09832, 09658, 08700, 08356, 07198, 09750, 03960, 03979, 08862, 06435, 04378, 00310, 09737, 07848, 04206, 08482, 09152, 08827, 08295, 03603, 08792, 05556, 07092, 06921, 06013, 03965, 03062, 09724, 04974, 08698, 00737, 09236, 06603, 04650	857
9	1	08000	2467
9	2	02088	2426
9	3	01261	2380
9	4	02772	2195
9	5	00679	2181
9	6	01950	2119

h	id	n	m
9	7	00025	2104
9	8	04848	2091
9	9	04931	2079
9	10	05959	2063
9	11	08997	2027
9	12	06276	2014
9	13	06503	1999
9	14	06265	1989
9	15	05315	1961
9	16	06171	1953
9	17	04407	1897
9	18	03256	1868
9	19	00814	1860
9	20	06393	1859
9	21	01302	1779
9	22	03242	1768
9	23	00886	1738
9	24	02034	1732
9	25	00264	1731
9	26	00250	1713
9	27	04340	1685
9	28	06428	1680
9	29	03092	1673
9	30	08995	1647
9	31	01512	1641
9	32	09050	1620
9	33	01308	1610
9	34	04393	1575
9	35	01003	1543
9	36	05419	1528
9	37	01092	1505
9	38	01634	1487
9	39	05142	1460
9	40	07680	1419
9	41	02729	1396
9	42	02130	1385
9	43	04608	1346
9	44	03267	1342
9	45	02798	1337
9	46	02437	1322
9	47	03456	1311

h	id	n	m
9	48	02855	1306
9	49	00291	1250
9	50	03260	1244
9	51	03015	1230
9	52	01154	1224
9	53	06267	1194
9	54	02149	1175
9	55	03167	1167
9	56	02089	1152
9	57	00525	1113
9	58	00739	1079
9	59	01086	1057
9	60	06971	1043
9	61	06172	1013
9	62	03664	985
9	63	00631	960
9	64	00617	914
9	65	02201	898
9	66	03084	890
9	67	05674	885
9	68	03295	874
9	69	02718	842
9	70	00875	827
9	71	01236	809
9	72	00731	798
9	73	02416	776
9	74	04260	761
9	75	01901	734
9	76	03193	718
9	77	00717	690
9	78	01152	680
9	79	09415	663
9	80	08291	643
9	81	02655	637
9	82	09481	625
9	83	07523	618
9	84	08456	602
9	85	06681	583
9	86	04071	570
9	87	03082	564
9	88	04355	558

h	id	n	m
9	89	07529	552
9	90	09153	545
9	91	00710	533
9	92	09889	519
9	93	00904	514
9	94	03079	506
9	95	05701	487
9	96	00432	477
9	97	06399	468
9	98	00552	461
9	99	04575	452
9	100	02060	445
9	101	01979	441
9	102	03436	434
9	103	09049	415
9	104	02119	412
9	105	07376	402
9	106	07972	395
9	107	05762	387
9	108	09341	370
9	109	09345	355
9	110	01552	340
9	111	02444	330
9	112	09825	316
9	113	02042	308
9	114	07135	295
9	115	04237	284
9	116	09609	267
9	117	08945	254
9	118	03857	228
9	119	08472	218
9	120	02803	201
9	121	06074	188
9	122	02167	178
9	123	05577	167
9	124	08934	154
9	125	02951	142
9	126	08757	132
9	127	08474	116
9	128	00471	103
9	129	06010	88

h	id	n	m
9	130	05006	76
9	131	00374	76
9	132	09452	73
9	133	08635	73
9	134	07729	73
9	135	08855	72
9	136	06549	72
9	137	00660	72
9	138	09295	71
9	139	00964	70
9	140	03361	69
9	141	08004	68
9	142	08733	67
9	143	07880	67
9	144	05102	67
9	145	08656	66
9	146	07581	66
9	147	07103	66
9	148	09461	65
9	149	09615	64
9	150	08278	64
9	151	06437	63
9	152	05329	63
9	153	03372	63
9	154	03801	61
9	155	09210	60
9	156	07611	60
9	157	07007	60
9	158	08529	58
9	159	07026	58
9	160	03997	58
9	161	09901	57
9	162	08848	57
9	163	03080	57
9	164	09605	56
9	165	09548	56
9	166	03720	56
9	167	01456	56
9	168	09628	55
9	169	09430	55
9	170	08044	55

h	id	n	m
9	171	06217	55
9	172	09907	54
9	173	09730	54
9	174	07450	54
9	175	04092	54
9	176	03930	54
9	177	03445	54
9	178	01646	54
9	179	09709	53
9	180	08053	53
9	181	02085	53
9	182	00684	53
9	183	09293	52
9	184	08670	52
9	185	07809	52
9	186	04976	52
9	187	08408	51
9	188	08338	51
9	189	07597	51
9	190	09396	50
9	191	01717	50
9	192	09156	49
9	193	00735	49
9	194	09716	48
9	195	07606	48
9	196	03710	48
9	197	02383	48
9	198	09751	47
9	199	08560	47
9	200	07745	47
9	201	06627	47
9	202	01116	47
9	203	07834	46
9	204	06459	46
9	205	09155	45
9	206	03933	45
9	207	09441	43
9	208	09362	43
9	209	05945	43
9	210	08224	42
9	211	07885	42

h	id	n	m
9	212	06384	42
9	213	05939	42
9	214	03675	42
9	215	02982	42
9	216	01727	42
9	217	09243	41
9	218	00311	41
9	219	06284	40
9	220	02620	40
9	221	04040	39
9	222	03969	39
9	223	03723	39
9	224	00087	39
9	225	02650	37
9	226	09432	36
9	227	08307	36
9	228	00135	36
9	229	09786	35
9	230	04159	35
9	231	04003	35
9	232	03697	35
9	233	08711	34
9	234	07287	34
9	235	02866	34
9	236	01825	34
9	237	06011	33
9	238	09906	32
9	239	08668	32
9	240	07158	32
9	241	03966	32
9	242	09342	31
9	243	08669	31
9	244	07390	30
9	245	05956	30
9	246	03907	30
9	247	09971	29
9	248	08819	29
9	249	07820	29
9	250	05942	29
9	251	03970	29
9	252	00058	29

h	id	n	m
9	253	07700	28
9	254	03967	28
9	255	07903	27
9	256	03654	27
9	257	02902	27
9	258	09618	26
9	259	07560	26
9	260	07178	26
9	261	05828	26
9	262	04210	26
9	263	02399	26
9	264	02045	26
9	265	08991	25
9	266	08434	25
9	267	05063	25
9	268	03797	25
9	269	03601	25
9	270	03350, 09354, 02513, 02401, 08181, 01733, 08632, 07528, 07402, 05635, 03973, 03968, 03903, 03781, 03661, 09611, 03612, 00197, 06324, 04720, 00361, 05170, 05092, 04628, 01737, 06982, 06915, 06023, 03977, 03368, 09694, 08962, 05341, 03642, 07425, 07115, 05514, 04219, 03122, 02038, 00885, 09613, 07587, 03927, 01467, 00309, 05123, 03862, 03760, 09929, 08386, 07419, 06168, 04365, 03780, 09612, 03306, 02984, 00837, 06613, 05456, 03232, 09607, 08852, 05513, 01590, 09592, 09537, 05512, 05205, 03381, 01145, 01066, 00690, 09353, 08832, 06995, 04761, 04760, 01809, 09429, 08686, 06907, 03912, 09330, 08808, 05358, 02877, 01742, 00319, 09033, 08928, 06497, 05661, 05375, 05125, 05088, 01997, 01063, 06719, 06544, 05192, 04948, 04705, 02461, 09474, 08951, 05454, 04293, 02050	1181

Selection Technique:

Systematic sampling is a suitable technique. For school h_i

- Calculate the sampling interval $k_h = MOS_{hi}/n_h$.
- Choose a random starting number between 1 and k_{hi}.
- Select the student at the random start position and every $k'_{hi}th$ student thereafter from the ordered roster.
- If schools are linked due to insufficient numbers, the rosters need to be combined and sampled uniformly.
- Record unresponsive students and report adjusted weights.

4. Write down the overall sampling fraction based on the stratified PPeS design, indicating the overall probability of inclusion for a given student, from a given school (or linked set of schools), in a given stratum. Be careful with notation. Keep in mind that the MOS values used for the sampled schools at the first stage and the denominator at the second stage (Did you sample a single school? Or a linked set of schools?) will depend on your response to Task #2 above

- The overall sampling fraction is $f = \frac{n}{N} = \frac{4{,}721}{830138} = .0057$
- The inclusion probability for a given student is $P_{hi} = \frac{n_h \times MOS_{hi}}{MOS_h} \times \frac{m_h}{MOS_{hi}} = \frac{n_h \times m_h}{MOS_h}$.

```
linked_schools<- linked_schools%>%
  group_by(Region)%>%
  mutate(P_h = n_h*total_mos/M_h,
         P_i=m_h_star/total_mos,
         Prob=P_h*P_i,
         epsem_check = abs(Prob - mean(Prob)) < 1e-6)

stopifnot(all(linked_schools$epsem_check))

# check for false
linked_schools |>
  distinct(m_h_star, n_h) |>
  reframe(Students = sum(m_h_star*n_h)) |>
  arrange(Region) |>
  kable()
```

Region	Students
1	20.25515
2	31.13640
3	49.09358
4	27.62118
5	107.54401
6	188.45655
7	1091.90130
8	1073.93843
9	2131.58628

SM 625: Week 11 Sampling Project Notes

By now, you should have noted from the sampling frame that one approach for sorting the schools within a region is by grade level of the schools (middle, generally including grades 7 and 8, and high, generally including grades 9 through 12). We would want to reduce the chance of a random sample of schools within a region only including students from grades 7 and 8 by sorting our list in this fashion.

This week, you have been provided with the actual classroom rosters from a randomly sampled middle school according to your design (see the file "sample_school_student_list.xls" on Canvas). Suppose that the randomly sampled middle school was from Region 7, and the MOS for this school was 242. At this point, you have determined the m_h needed from Region 7 to maintain epsem overall (see last week's project notes). Given the actual classroom rosters, what is the actual size of this school? Assuming that this school was not linked with any other schools, what is the sampling rate that you would apply to this school to achieve epsem? And what would your expected actual sample size be, once you apply this rate to the actual roster?

Given your plan for within-school sampling developed last week, describe your approach to selecting the sample at your specified rate, and then implement that technique to actually select the sample. You can provide the resulting sample as an Appendix for your final project, but the selection technique needs to be clearly described in the body of your report. Ultimately, your description of this process should enable readers to understand what would happen to select the sample of students within each sampled school.

- Selection Technique: Systematic sampling is a suitable technique. For school hi: • Calculate the sampling interval $k_h i = Mos_{hi}/n_h$ Choose a random starting number between 1 and k_hi.
- Select the student at the random start position and every k_hi-th student thereafter from the ordered roster.
- If schools are linked due to insufficient numbers, the rosters need to be combined and sampled uniformly.
- Record unresponsive students and report adjusted weights.

The overall sampling fraction is

$$f = \frac{n}{N} = \frac{4,721}{830138} = .0057$$

The inclusion probability for a given student is

$$P_{hi} = \frac{a_h \times MOS_{hi}}{MOS_h} \times \frac{m_h}{MOS_{hi}} = \frac{a_h \times m_h}{MOS_h}$$

The number of students to sample from this school (based on MOS) is:

$$m_{hi} = f \cdot M_{hi} = 0.0057 \cdot 242$$

$$\text{Sampling Rate} = \frac{m_{hi}}{N_{hi}} = \frac{2}{219}$$

The actual size is 219. The sampling rate should be 0.07762557. The expected actual expected sample size is 17.

```
# Step 1:  Calculate how many students to sample
f_overall <- 0.0057
oneschool <- sample_selected
MOS_7 <- 242
ACT_MOS_7 <- nrow(oneschool)
M_h_7 <- 191992
m_h_start_7 <- ceiling(16.29896)
# Step 2: Sampling rate
sam_rate <- m_h_start_7/ACT_MOS_7


# Step 2: Calculate sampling interval
k_interval <- ACT_MOS_7 / m_h_start_7
round_k_interval <- k_interval*100000
round_mos <- 219*100000+99999



# Step 3: Random start between 1 and interval
set.seed(123)
start <- sample(1:round_k_interval, 1)

# Step 4: Select every `interval`-th student starting from `start`
indices <- seq(start, by = round_k_interval, length.out = m_h_start_7)
true_indices <- floor(indices/100000)
sampled_students <- oneschool[true_indices, ]

# View sampled students
sampled_students |>
  kable()
```

BCODE	NAME	DCODE	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	SECU_MOS
09139	Bromley Elementary School	17130	Bromley Area Schools	4	17	Chippewa	Public		0	1	0	0	0	1	1	0	0	1	1	1	0	0	1	4491	449
03284	Ring Lardner Middle School	11300	Niles Community School District	7	11	Berrien	Public	272	294	0	0	0	0	566	539	1	1	33	93784	93219	1	1	566	56633	566
05790	Pearl Alternative/Adult Education School	03050	Fennville Public Schools	7	03	Allegan	Public		0	17	29	9	5	60	193	0	0	1	4474	4415	0	0	60	5961	5961
09471	Fruitport Alternative High School	61080	Fruitport Community Schools	7	61	Muskegon	Public		0	1	10	9	14	34	138	0	0	1	1979	1946	0	0	34	5961	5961
03945	ST PATRICK SCHOOL	41170	NA	7	41	Kent	nonpublic	9	8	0	0	0	0	17	84	0	0	1	656	640	0	0	17	5961	5961
06332	EAU CLAIRE SDA SCHOOL	11250	NA	7	70	Ottawa	nonpublic	1	4	0	0	0	0	5	30	0	0	1	77	73	0	0	5	5961	5961
02777	Northwestern High School	25010	Flint City School District	8	25	Genesee	Public		0	384	195	213	122	914	498	1	1	42	119127	118214	1	1	914	91442	914

BCODE	NAME																									SECU_MOS
03666	ST CLARE OF MONTEFALCO SCH	82055	NA	8	33	Ingham	nonpublic	30	30	0	0	0	0	60	145	0	0	1	3631	3572	0	0	60	4583	1	4583
03846	ST LUKE LUTHERAN SCHOOL	50070	NA	8	46	Lenawee	nonpublic	11	19	0	0	0	0	30	91	0	0	1	1397	1368	0	0	30	4583	1	4583
09725	Special Services - North	25000	Genesee ISD	8	25	Genesee	Public	4	3	1	3	2	0	13	37	0	0	1	265	253	0	0	13	4583	1	4583
00804	Cousino Senior High School	50230	NA	9	50	Macomb	Public		0	472	489	475	424	1860	898	1	1	112	3226	3208	112	1	1860	1860	112	1860
02416	Marysville High School	74100	Marysville Public Schools	9	74	St. Clair	Public		0	199	216	172	189	776	774	1	1	58	1691	1634	108	1	776	776	58	776
08474	Compass Point	50040	Anchor Bay School District	9	50	Macomb	Public		5	7	17	46	41	116	337	1	1	4	1598	1586	15	1	116	116	4	116
00684	CHRIST THE GOOD SHEPHERD SCH	82090	NA	9	63	Oakland	nonpublic	19	34	0	0	0	0	53	198	0	0	1	4495	4443	0	0	53	7708	1	7708
01825	CHRIST THE KING LUTH-SEBEWAI	79145	NA	9	74	St. Clair	nonpublic	18	16	0	0	0	0	34	144	0	0	1	2146	2113	0	0	34	7708	1	7708

BCODE	BNAME	DCODE	District_Name	region	County_ID	County_Name	[illegible]	g7	g8	g9	g10	g11	g12	tot_all	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	SECU	SECU_MOS
00361	BORCULO CHRISTIAN SCHOOL	70350	[illegible]	9	63	Oakland	nonpublic	8	10	0	0	0	0	18	90	0	0	1	770	753	0	0	18	7708	1	7708
09353	Bridges Assessment Center	79000	Tuscola ISD	9	79	Tuscola	Public	0	0	3	0	1	2	6	36	0	0	1	124	119	0	0	6	7708	1	7708

Week 13

1. Based on the final sample design that your team has developed, formulate a sampling error calculation model that users of your data will be able to employ to estimate sampling variance. That is, what stratum codes will you provide to users? How will you form sampling error computation units (SECUs)? How many SECUs will there be per stratum? What are expected sample sizes per SECU?

```
# Week 13 - Q1: Sampling Error Calculation Model


# Inputs based on project design
a_select <- n_opt # n_opt
b_star <- m_opt # m_opt
n_strata_var <- a_select / 2
n_regions <- 9

school_id <- 1:a_select

var_stratum_id <- rep(1:n_strata_var, each = 2)
# drop last
var_stratum_id <- var_stratum_id[1:length(school_id)]

SECU_id <-  1:a_select

expected_sample_size <- b_star

# Each pair of schools forms one variance stratum
# Each school is one SECU
```

```r
variance_strata <- tibble(
  school_id,
  var_stratum_id,
  SECU_id,
  expected_sample_size,
)

# Summary table for documentation
summary_table <- tibble(
  Description = c(
    "Total Schools Selected (a_select)",
    "Explicit Strata (Regions)",
    "Variance Estimation Strata (Paired PSUs)",
    "SECUs per Variance Stratum",
    "Expected Sample Size per SECU (b*)"
  ),
  Value = c(
    a_select,
    n_regions,
    n_strata_var,
    2,
    b_star
  )
)

kable(summary_table, align = "lc")
```

Description	Value
Total Schools Selected (a_select)	293.22953
Explicit Strata (Regions)	9.00000
Variance Estimation Strata (Paired PSUs)	146.61476
SECUs per Variance Stratum	2.00000
Expected Sample Size per SECU (b*)	76.68084

```r
kable(head(variance_strata))
```

school_id	var_stratum_id	SECU_id	expected_sample_size
1	1	1	76.68084
2	1	2	76.68084

school_id	var_stratum_id	SECU_id	expected_sample_size
3	2	3	76.68084
4	2	4	76.68084
5	3	5	76.68084
6	3	6	76.68084

```
#kable(variance_strata)
```

2. Describe the variance estimation procedures that one would employ to form a confidence interval for one of the three key descriptive parameters. This should build on your proposed SECUs from the first task. How many degrees of freedom will your sampling error calculation model afford? In addition, write the formula for one of the estimated proportions or means; are weights necessary in forming this estimator, given your sample design? That is, is your design epsem, or will weights be needed to compensate for unequal probabilities of selection?

```
df <- a_select / 2           # Degrees of freedom = number of variance strata
num_strata <- df


# Simulate SECU-level estimates for a descriptive proportion (e.g., smoked a cigarette)
set.seed(9999)
cig_lower <- MI_school_samples_table |>
  slice(1) |>
  pull(lower)

cig_upper <- MI_school_samples_table |>
  slice(1) |>
  pull(upper)

p_secu_1 <- runif(num_strata, cig_lower, cig_upper)  # SECU 1 estimates

mj_lower <- MI_school_samples_table |>
  slice(2) |>
  pull(lower)

mj_upper <- MI_school_samples_table |>
  slice(2) |>
  pull(upper)
```

```
p_secu_2 <- runif(num_strata, mj_lower, mj_upper)  # SECU 2 estimates


# Paired Difference Variance Estimation
diffs <- p_secu_1 - p_secu_2
var_estimate <- mean(diffs^2) / 2         # Variance across pairs
se_estimate <- sqrt(var_estimate)

# Confidence interval (95%)
t_crit <- qt(0.975, df = df)
estimate_mean <- mean(c(p_secu_1, p_secu_2))
CI_lower <- estimate_mean - t_crit * se_estimate
CI_upper <- estimate_mean + t_crit * se_estimate
```

Degrees of Freedom (df): 146.6148

Standard Error (SE): 0.07179

95% Confidence Interval for estimated proportion:

[0.0575 , 0.3413]

Estimator Formula for Proportion:

$\hat{p}$ = sum(w_hij * y_hij) / sum(w_hij)

where:

y_hij = 1 if student j in school i of stratum h has the trait (e.g., smoked), 0 otherwise

w_hij = final weight for student hij (includes selection probability, nonresponse, etc.)

Are weights needed? YES.

1. Although the design aimed for EPSEM, weights are necessary in practice.

2. Adjustments are needed for:

   - School-level nonresponse (30%)

   - Student-level nonresponse (70%)

3. Weights also adjust for second-stage linking or other deviations during implementation.

3. Keep in mind the client's request for estimates and inference related to a 20% subclass. Will confidence intervals for the subclass be formed in the same way? Are your SECUs large enough to accommodate this request?

```r
total_secus <- a_select            # Number of SECUs (schools)
expected_b_star <- b_star          # Expected completes per SECU
subclass_pct <- 0.20          # Subclass proportion (20%)
df <- total_secus / 2         # Degrees of freedom remains the same

# Estimate expected subclass size per SECU
expected_subclass_per_secu <- expected_b_star * subclass_pct
```

```
 Subclass Estimation for 20% Group

 Confidence Intervals:

CI for subclass estimates can be formed using the same paired difference method.

Degrees of Freedom remains: 146.6148

 SECU Size Check:

Expected sample size per SECU (b*): 76.68084

Estimated subclass members per SECU: 15.3

 This is generally sufficient for stable variance estimation at the subclass level.
```