SURV625 Applied Sampling

2025-04-22

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) |>
```

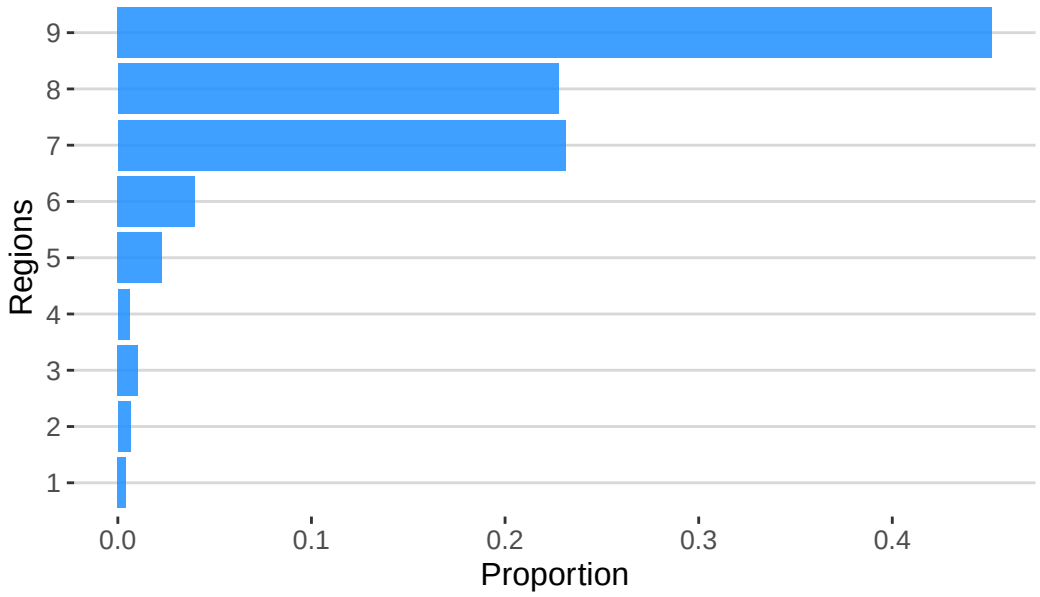
```
  reframe(M_h = sum(tot_all), # total of students in stratum
          N_h = n()) |> # total of schools in stratum
  mutate(prop_allocation = M_h/sum(M_h))

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

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

```
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.

```
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),
```

```
          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 $C = \$500,000$.
- 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.

```
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,
```

```
         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,
         # recalcualte 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
```

```r
# Compute proportional allocation of clusters to each stratum
region_summary <- strata_Prop_allocate |>
  # Ensure at least 1 cluster per
  mutate(n_h = n_opt * prop_allocation) |>
  mutate(N_h = as.double(N_h)) |>
  group_by(Region) |>
  reframe(across(where(is.double), ~ sum(.x))) |>
  mutate(k_h =  M_h / round(n_h)) |>    # zone size
  # create random start values
  rowwise() |>
  mutate(RN = sample(1:k_h, 1)) |>
  ungroup()

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

Region	prop_allocation	n_h	k_h	RN
1	0.0042896	1.257851	3561.000	3168
2	0.0065941	1.933580	2737.000	2310
3	0.0103971	3.048727	2877.000	1321
4	0.0058484	1.714931	2427.500	131
5	0.0227757	6.678517	2701.000	2122
6	0.0399126	11.703565	2761.083	2114
7	0.2312772	67.817307	2823.412	374
8	0.2274682	66.700394	2818.358	380
9	0.4514370	132.374656	2839.053	1673

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

```r
# 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
```

```r
# 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
})
```

```r
# 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) )
```

```r
# 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"), ~
                         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) |>
      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)

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

```
# Form Pseudo-Strata for Paired Selection Model
pseudo_strata_df <- sample_selected |>
```

```r
  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	3561.000	3168	20.25515
2	2737.000	2310	15.56820
3	2877.000	1321	16.36453
4	2427.500	131	13.80775
5	2701.000	2122	15.36343
6	2761.083	2114	15.70519
7	2823.412	374	16.05971
8	2818.358	380	16.03097
9	2839.053	1673	16.14868

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")
  )
```

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

```r
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	25.47798
2	30.10236
3	49.89097
4	23.67933
5	102.60493
6	183.80667
7	1089.12657
8	1069.27197
9	2137.67629

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

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

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

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

$$\text{Sampling Rate} = \frac{m_7^*}{MOS_7} = \frac{16.05737}{242}$$

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 <- read_csv("~/work/d/SURV625project/data/schoolframe.csv")
MOS_7 <- 242
ACT_MOS_7 <- nrow(oneschool)
m_h_start_7 <- 16.05737

# Step 2: Sampling rate
sam_rate <- m_h_start_7/MOS_7

# Step 3: Expected sample size
m_exp <- sam_rate * ACT_MOS_7
m_exp_pra <- ceiling(m_exp)
# Step 2: Calculate sampling interval
k_interval <- ACT_MOS_7 / m_exp_pra
round_k_interval <- k_interval*10
round_mos <- 219*10+9



# 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_exp_pra)
true_indices <- floor(indices/10)
sampled_students <- oneschool[true_indices, ]

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

Table 16: Sampled Students

ID	Grade	class	Firstname	Lastname
1	7	Grady Vest	MAGEE	MONICA L
16	7	Grady Vest	SCHWARTZ	DAVID SCOTT
30	7	Qixuan Li	RAYNOR	GREGORY K
45	7	Qixuan Li	FRANKE	MIRA A
59	7	Bill Pesau	BLACK	STEPHEN P
74	7	Andrew Bellman	MARBUT	JOANNE RENEE
89	7	Andrew Bellman	TRECKER	MOLLY A
103	8	Joe Williams	DAWSON	REBECCA S
118	8	Joe Williams	KROSKY	PAULA MICHELE
132	8	Robert McFay	O BRIEN	ERIN TERESE
147	8	Joseph Miden	KOSOVE	DANIEL BRIAN
162	8	Joseph Miden	LOEVY	DEBRA L
176	8	James Vogner	DWORZANOWSKI	GREGORY WILLIAM
191	8	James Vogner	GATTO	JULIA LYNN
205	8	Jane Doe	KAYE	PETER MITCHELL

ID	Grade	class	Firstname	Lastname
1	7	Grady Vest	MAGEE	MONICA L
16	7	Grady Vest	SCHWARTZ	DAVID SCOTT
30	7	Qixuan Li	RAYNOR	GREGORY K
45	7	Qixuan Li	FRANKE	MIRA A
59	7	Bill Pesau	BLACK	STEPHEN P
74	7	Andrew Bellman	MARBUT	JOANNE RENEE
89	7	Andrew Bellman	TRECKER	MOLLY A
103	8	Joe Williams	DAWSON	REBECCA S
118	8	Joe Williams	KROSKY	PAULA MICHELE
132	8	Robert McFay	O BRIEN	ERIN TERESE
147	8	Joseph Miden	KOSOVE	DANIEL BRIAN
162	8	Joseph Miden	LOEVY	DEBRA L
176	8	James Vogner	DWORZANOWSKI	GREGORY WILLIAM
191	8	James Vogner	GATTO	JULIA LYNN
205	8	Jane Doe	KAYE	PETER MITCHELL


```
#write.csv(sampled_students, "sampled_students.csv", row.names = FALSE)
kbl(sampled_students, caption = "Sampled Students") %>%
  kable_styling() #%>%
```

```
  #save_kable("sampled_students_table.png")
```

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
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

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

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

```r
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?

```
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.
```