Tidy Survey Analysis in R using the srvyr Package

Master in Computational Social Science

Patrick Kraft, IPP-CSIC

November 2025

Introduction

Overview

- At the end of this workshop series, you will be able to
    - Specify a survey design in R to create a survey object
    - Calculate point estimates and their standard errors with survey data
        - Means, quantiles, and ratios
        - Proportions, totals, and counts
    - Perform t-tests
    - Fit regression models
- Main resources:
    - Stephanie A. Zimmer, Rebecca J. Powell, and Isabella C. Velásquez. 2024. Exploring Complex Survey Data Analysis Using R. CRC Press
    - Online documentation for the sampling package

Acknowledgments

This workshop is based on material for the course "Tidy Survey Analysis in R" taught by Stephanie Zimmer, Rebecca Powell, and Isabella Velásquez at the conference of the American Association for Public Opinion Research (AAPOR).

Specifying sample design objects

Review of sampling designs

- **Simple random sampling:** every unit has the same chance of being selected
    - Without replacement: units can only be selected once
    - With replacement: units can be selected more than once
- **Systematic sampling:** sample n individuals from an ordered list and sampling individuals at an interval with a random starting point
- **Probability proportional to size:** probability of selection is proportional to "size"
- **Stratified sampling:** divide population into mutually exclusive subgroups (strata). Randomly sample within each stratum
- **Clustered sampling:** divide population into mutually exclusive subgroups (clusters). Randomly sample clusters and then individuals within clusters

Create Your Own Probability Sample

The `sampling` package provides a set of useful functions for...

- **Sampling:** Stratification, two-stage, unequal probabilities, balanced sampling
- **Estimation:** calibration and regression estimator
- **Tools:** computation of inclusion probabilities, crossing strata
- **Data bases:** Swiss municipalities, Belgian municipalities.

Install and load the package

```
install.packages(setdiff("sampling", rownames(installed.packages())))
library(sampling)
```

Example: Simple random sampling w/o replacement

Select a sample of size 10 out of a population of 50:

```
s <- srswor(
   n = 10,  ## Sample size
   N = 50   ## Population size
)

matrix(s, nrow = 5, byrow = TRUE)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]    0    0    1    0    0    0    1    0    0     0
## [2,]    0    1    0    0    0    0    0    0    0     0
## [3,]    0    0    0    1    0    0    0    0    0     0
## [4,]    0    0    0    0    0    0    0    1    1     1
## [5,]    0    0    1    0    0    1    0    1    0     0
```

Here are the selected sample IDs:

```
which(s==1)
```

```
##  [1]  3  7 12 24 38 39 40 43 46 48
```

Example: Stratified sampling

Variable of stratification (3 strata):

```
group <- rep(1:3, each = 5)
```

Matrix of balancing variables:

```
X <- cbind(1:15)
```

Vector of inclusion probabilities (sample size of 9):

```
pik <- rep(3/5, times = 15)
```

Selection of a stratified sample

```
s <- balancedstratification(
   X = X, strata = group, pik = pik,
   comment = FALSE
)
```

The sample is:

```
matrix(s, nrow = 3, byrow = TRUE)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    1    1    1    0    0
## [2,]    0    1    1    1    0
## [3,]    0    1    1    0    1
```

```
which(s==1)
```

```
## [1]  1  2  3  7  8  9 12 13 15
```

Example: Clustered sampling

Selection of 2 clusters:

```
s <- balancedcluster(
   X = X, m = 2, cluster = group,
   selection = 2, # cluster selection equal or by size
   comment = FALSE
)
```

The sample of clusters with the inclusion probabilities of the clusters:

```
head(s)
```

```
##      [,1]      [,2]
## [1,]    1 0.6666667
## [2,]    1 0.6666667
## [3,]    1 0.6666667
## [4,]    1 0.6666667
## [5,]    1 0.6666667
## [6,]    1 0.6666667
```

The selected clusters:

```
unique(group[s[,1]==1])
```

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

The selected units:

```
which(s[,1]==1)
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

Overview of Survey Analysis using `srvyr` Package

1. Create a `tbl_svy` object using: `as_survey_design` or `as_survey_rep`
2. Subset data (if needed) using `filter` (subpopulations)
3. Specify domains of analysis using `group_by`
4. Within `summarize`, specify variables to calculate including means, totals, proportions, quantiles and more

Set-up for Analysis

- `srvyr` package uses tidy-syntax but uses the `survey` package behind it to do calculations
- Install both packages (and a few others):

```
# Install survey and srvyr packages (only once!)
install.packages(setdiff(c("survey", "srvyr", "tidyverse", "here"),
                         rownames(installed.packages())))

# Load packages
library(tidyverse) # for tidyverse
library(here) # for file paths
library(survey) # for survey analysis
library(srvyr) # for tidy survey analysis
```

Determining the design

- Look at documentation associated with the analysis file
- Keywords to look for: methodology, design, analysis guide, technical documentation
- Documentation will indicate the variables needed to specify the design. Look for:
    - weight (almost always)
    - strata and/or clusters/PSUs. Sometimes pseudo-strata and pseudo-cluster OR
    - replicate weights (this is used instead of strata/clusters for analysis)
    - might also see finite population correction or population sizes
- Documentation may include syntax for SAS, SUDAAN, Stata and/or R!

Survey Datasets

American National Election Study (ANES) 2020

- Pre and post election surveys
- Fielded almost every 2 years since 1948
- Topics include voter registration status, candidate preference, opinions on country and government, party and ideology affiliation, opinions on policy, news sources, and more
- Collaboration of Stanford, University of Michigan – funding by the National Science Foundation
- **Target Population**: US citizens, 18 and older living in US
- **Mode**: Web, videoconference, or telephone.
- **Sample Information**: Pseudo-strata and pseudo-cluster included for variance estimation

American National Election Study (ANES) 2020

- https://electionstudies.org/data-center/2020-time-series-study/
- Opened the file "User Guide and Codebook"
- Section "Data Analysis, Weights, and Variance Estimation": Page 8-12 includes information on weights and strata/cluster variables

> For analysis of the complete set of cases using pre-election data only, including all cases and representative of the 2020 electorate, use the full sample pre-election weight, V200010a. For analysis including post-election data for the complete set of participants (i.e., analysis of post-election data only or a combination of pre- and post-election data), use the full sample post-election weight, V200010b. Additional weights are provided for analysis of subsets of the data...

For weight	Use variance unit/PSU/cluster	and use variance stratum
V200010a	V200010c	V200010d
V200010b	V200010c	V200010d

Residential Energy Consumption Survey (RECS) 2015

- Energy consumption/expenditures collected through energy suppliers
- Fielded 14 times between 1950 and 2015
- Topics include appliances, electronics, heating, a/c, temperatures, water heating, lighting, energy bills, respondent demographics, and energy assistance
- Funded by the Energy Information Administration
- **Target Population**: Primary occupied housing units in the US
- **Mode**: In-person, paper, and web interview mode
- **Sample Information**: BRR Replicate weights included for variance estimation

https://www.eia.gov/consumption/residential/index.php

Residential Energy Consumption Survey (RECS) 2015

- https://www.eia.gov/consumption/residential/data/2015/index.php?view=microdata
- Opened the file "Using the 2015 microdata file to compute estimates and standard errors (RSEs)"
- Page 4:

> The following instructions are examples for calculating any RECS estimate using the final weights (NWEIGHT) and the associated RSE using the replicate weights (BRRWT1 – BRRWT96).
>
> Let ϵ be the Fay coefficent ... and $\epsilon = 0.5$

- Page 9: Syntax given for survey package which is similar to srvyr (as we will see)

```
library(survey)
RECS15 <- read.csv(file='< location where file is stored >', header=TRUE, sep=",")
sampweights <- RECS15$NWEIGHT
brrwts <- RECS15[, grepl("^BRRWT", names(RECS15))]
des <- svrepdesign(weights=sampweights, repweights=brrwts, type="Fay",
                   rho=0.5, mse=TRUE, data=RECS15)
```

Specify the sampling design

- This creates a `tbl_svy` object that then correctly calculates weighted estimates and SEs.

```
as_survey_design(
   .data,
   ids = NULL,#cluster IDs/PSUs
   strata = NULL,#strata variables
   variables = NULL,#defaults to all in .data
   fpc = NULL,#variables defining the finite population correct
   nest = FALSE,#TRUE/FALSE - relabel clusters to nest within strata
   check_strata = !nest, #check that clusters are nested in strata
   weights = NULL,# weight variable
   ...
)
```

Syntax for common designs

```
# simple random sample (SRS)
apisrs |> as_survey_design(fpc = fpc)

# stratified sample
apistrat |> as_survey_design(strata = stype, weights = pw)

# one-stage cluster sample
apiclus1 |> as_survey_design(ids = dnum, weights = pw, fpc = fpc)

# two-stage cluster sample, weights computed from pop size
apiclus2 |> as_survey_design(ids = c(dnum, snum), fpc = c(fpc1, fpc2))

# stratified, cluster design
apistrat |> as_survey_design(ids = dnum, strata = stype, weights =pw, nest = TRUE)
```

- examples from `srvyr` help documentation

ANES Example

For weight	Use variance unit/PSU/cluster	and use variance stratum
V200010b	V200010c	V200010d

```
options(width=130)
anes <- read_rds(here("Data", "anes_2020.rds")) |>
   mutate(Weight=V200010b/sum(V200010b)*231592693)
   # adjust weight to sum to citizen pop, 18+ in Nov 2020 per ANES methodology documentation

anes_des <- anes |>
   as_survey_design(weights = Weight,
                    strata = V200010d,
                    ids = V200010c,
                    nest = TRUE)
summary(anes_des)
```

ANES Example (cont'd)

```
## Stratified 1 - level Cluster Sampling design (with replacement)
## With (101) clusters.
## Called via srvyr
## Probabilities:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 4.839e-06 2.657e-05 4.689e-05 7.688e-05 8.331e-05 3.895e-03
## Stratum Sizes:
##              1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29
## obs        167 148 158 151 147 172 163 159 160 159 137 179 148 160 159 148 158 156 154 144 170 146 165 147 169 165 172 133 157
## design.PSU   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## actual.PSU   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##             30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50
## obs        167 154 143 143 124 138 130 136 145 140 125 158 146 130 126 126 135 133 140 133 130
## design.PSU   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## actual.PSU   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## Data variables:
##  [1] "V200010b"                "V200010d"                "V200010c"                "V200002"
##  [5] "V201006"                 "V201102"                 "V201101"                 "V201103"
##  [9] "V201025x"                "V201231x"                "V201233"                 "V201237"
## [13] "V201507x"                "V201510"                 "V201549x"                "V201600"
## [17] "V201617x"                "V202066"                 "V202109x"                "V202072"
## [21] "V202073"                 "V202110x"                "InterviewMode"           "Weight"
## [25] "Stratum"                 "VarUnit"                 "Age"                     "AgeGroup"
## [29] "Gender"                  "RaceEth"                 "PartyID"                 "Education"
## [33] "Income"                  "Income7"                 "CampaignInterest"        "TrustGovernment"
## [37] "TrustPeople"             "VotedPres2016"           "VotedPres2016_selection" "VotedPres2020"
## [41] "VotedPres2020_selection" "EarlyVote2020"
```

RECS Example

- Final weights: NWEIGHT Replicate weights: BRRWT1 – BRRWT96

```
options(width=130)
recs <- read_rds(here("Data", "recs.rds"))

recs_des <- recs |>
   as_survey_rep(weights=NWEIGHT,
                 repweights=starts_with("BRRWT"),
                 type="Fay",
                 rho=0.5,
                 mse=TRUE)

summary(recs_des)
```

RECS Example (cont'd)

```
## Call: Called via srvyr
## Fay's variance method (rho= 0.5 ) with 96 replicates and MSE variances.
## Sampling variables:
##   - repweights: `BRRWT1 + BRRWT2 + BRRWT3 + BRRWT4 + BRRWT5 + BRRWT6 + BRRWT7 + BRRWT8 + BRRWT9 + BRRWT10 + BRRWT11 + BRRWT12 +
##     BRRWT13 + BRRWT14 + BRRWT15 + BRRWT16 + BRRWT17 + BRRWT18 + BRRWT19 + BRRWT20 + BRRWT21 + BRRWT22 + BRRWT23 + BRRWT24 +
##     BRRWT25 + BRRWT26 + BRRWT27 + BRRWT28 + BRRWT29 + BRRWT30 + BRRWT31 + BRRWT32 + BRRWT33 + BRRWT34 + BRRWT35 + BRRWT36 +
##     BRRWT37 + BRRWT38 + BRRWT39 + BRRWT40 + BRRWT41 + BRRWT42 + BRRWT43 + BRRWT44 + BRRWT45 + BRRWT46 + BRRWT47 + BRRWT48 +
##     BRRWT49 + BRRWT50 + BRRWT51 + BRRWT52 + BRRWT53 + BRRWT54 + BRRWT55 + BRRWT56 + BRRWT57 + BRRWT58 + BRRWT59 + BRRWT60 +
##     BRRWT61 + BRRWT62 + BRRWT63 + BRRWT64 + BRRWT65 + BRRWT66 + BRRWT67 + BRRWT68 + BRRWT69 + BRRWT70 + BRRWT71 + BRRWT72 +
##     BRRWT73 + BRRWT74 + BRRWT75 + BRRWT76 + BRRWT77 + BRRWT78 + BRRWT79 + BRRWT80 + BRRWT81 + BRRWT82 + BRRWT83 + BRRWT84 +
##     BRRWT85 + BRRWT86 + BRRWT87 + BRRWT88 + BRRWT89 + BRRWT90 + BRRWT91 + BRRWT92 + BRRWT93 + BRRWT94 + BRRWT95 + BRRWT96`
##   - weights: NWEIGHT
## Data variables:
##   - DOEID (dbl), Region (fct), Division (fct), MSAStatus (fct), Urbanicity (fct), HousingUnitType (fct), YearMade (ord),
##     SpaceHeatingUsed (lgl), HeatingBehavior (fct), WinterTempDay (dbl), WinterTempAway (dbl), WinterTempNight (dbl), ACUsed
##     (lgl), ACBehavior (fct), SummerTempDay (dbl), SummerTempAway (dbl), SummerTempNight (dbl), TOTCSQFT (dbl), TOTHSQFT (dbl),
##     TOTSQFT_EN (dbl), TOTUCSQFT (dbl), TOTUSQFT (dbl), NWEIGHT (dbl), BRRWT1 (dbl), BRRWT2 (dbl), BRRWT3 (dbl), BRRWT4 (dbl),
##     BRRWT5 (dbl), BRRWT6 (dbl), BRRWT7 (dbl), BRRWT8 (dbl), BRRWT9 (dbl), BRRWT10 (dbl), BRRWT11 (dbl), BRRWT12 (dbl), BRRWT13
##     (dbl), BRRWT14 (dbl), BRRWT15 (dbl), BRRWT16 (dbl), BRRWT17 (dbl), BRRWT18 (dbl), BRRWT19 (dbl), BRRWT20 (dbl), BRRWT21
##     (dbl), BRRWT22 (dbl), BRRWT23 (dbl), BRRWT24 (dbl), BRRWT25 (dbl), BRRWT26 (dbl), BRRWT27 (dbl), BRRWT28 (dbl), BRRWT29
##     (dbl), BRRWT30 (dbl), BRRWT31 (dbl), BRRWT32 (dbl), BRRWT33 (dbl), BRRWT34 (dbl), BRRWT35 (dbl), BRRWT36 (dbl), BRRWT37
##     (dbl), BRRWT38 (dbl), BRRWT39 (dbl), BRRWT40 (dbl), BRRWT41 (dbl), BRRWT42 (dbl), BRRWT43 (dbl), BRRWT44 (dbl), BRRWT45
##     (dbl), BRRWT46 (dbl), BRRWT47 (dbl), BRRWT48 (dbl), BRRWT49 (dbl), BRRWT50 (dbl), BRRWT51 (dbl), BRRWT52 (dbl), BRRWT53
##     (dbl), BRRWT54 (dbl), BRRWT55 (dbl), BRRWT56 (dbl), BRRWT57 (dbl), BRRWT58 (dbl), BRRWT59 (dbl), BRRWT60 (dbl), BRRWT61
##     (dbl), BRRWT62 (dbl), BRRWT63 (dbl), BRRWT64 (dbl), BRRWT65 (dbl), BRRWT66 (dbl), BRRWT67 (dbl), BRRWT68 (dbl), BRRWT69
##     (dbl), BRRWT70 (dbl), BRRWT71 (dbl), BRRWT72 (dbl), BRRWT73 (dbl), BRRWT74 (dbl), BRRWT75 (dbl), BRRWT76 (dbl), BRRWT77
##     (dbl), BRRWT78 (dbl), BRRWT79 (dbl), BRRWT80 (dbl), BRRWT81 (dbl), BRRWT82 (dbl), BRRWT83 (dbl), BRRWT84 (dbl), BRRWT85
##     (dbl), BRRWT86 (dbl), BRRWT87 (dbl), BRRWT88 (dbl), BRRWT89 (dbl), BRRWT90 (dbl), BRRWT91 (dbl), BRRWT92 (dbl), BRRWT93
##     (dbl), BRRWT94 (dbl), BRRWT95 (dbl), BRRWT96 (dbl), CDD30YR (dbl), CDD65 (dbl), CDD80 (dbl), ClimateRegion_BA (fct),
##     ClimateRegion_IECC (fct), HDD30YR (dbl), HDD65 (dbl), HDD50 (dbl), GNDHDD65 (dbl), BTUEL (dbl), DOLLAREL (dbl), BTUNG (dbl),
```

Continuous Dependent Variables

Weighted Analysis for Continuous Variables

- Common functions for continuous summaries
    - survey_mean
    - survey_total (like sum)
    - survey_median
    - survey_quantile
    - survey_ratio
- Always call within `summarize`/`summarise`

`survey_mean` Syntax

```
survey_mean(
  x,
  na.rm = FALSE,
  vartype = c("se", "ci", "var", "cv"),
  level = 0.95,
  proportion = FALSE,
  deff = FALSE,
  df = NULL,
  ...
)
```

To calculate a survey mean, we use this in `summarize`/`summarise`

```
survey_design_object |>
   summarize(
      mean_varname=survey_mean(x = continuous_varname)
      )
```

`survey_mean` Example 1: On average, how much do US households spend on energy each year?

This is an example using the `recs_des` survey design object and `survey_mean` function defaults

```
recs_des |>
   summarize(
      TD_mean=survey_mean(x = TOTALDOL)
      )
```

```
## # A tibble: 1 × 2
##   TD_mean TD_mean_se
##     <dbl>      <dbl>
## 1   1859.       15.6
```

`survey_mean` Example 2: What is the average temperature US households set their homes to on a summer day?

Run this code. What happens?

```
recs_des |>
   summarize(
      TD_mean=survey_mean(x = SummerTempDay)
      )
```

`survey_mean` Example 2: What is the average temperature US households set their homes to on a summer day?

Run this code. What happens?

```
recs_des |>
   summarize(
      TD_mean=survey_mean(x = SummerTempDay)
      )
```

```
## Error in `dplyr::summarise()`:
## ℹ In argument: `TD_mean = survey_mean(x = SummerTempDay)`.
## Caused by error in `svrVar()`:
## ! All replicates contained NAs
```

How do we fix this code?

`survey_mean` Example 2: Missing data solution

```
recs_des |>
   summarize(
      TD_mean = survey_mean(
        x = SummerTempDay,
        na.rm = TRUE )
      )
```

```
## # A tibble: 1 × 2
##   TD_mean TD_mean_se
##     <dbl>      <dbl>
## 1    72.4     0.0793
```

`survey_median` Syntax

```
survey_median(
  x,
  na.rm = FALSE,
  vartype = c("se", "ci"),
  level = 0.95,
  df = NULL,
  ...
)
```

`survey_median` Example: What is the median temperature US households set their homes to on a summer day?

```
recs_des |>
   summarize(
      TD_median=survey_median(x=_________,
                              na.rm=_________)
      )
```

```
recs_des |>
 summarize(
   TD_median=survey_median(x=SummerTempDay,
                           na.rm=TRUE)
 )
```

```
## # A tibble: 1 × 2
##   TD_median TD_median_se
##       <dbl>        <dbl>
## 1        72        0.252
```

`survey_ratio` Syntax

- Note this estimates: $\sum x_i / \sum y_i$ not $\sum \frac{x_i}{y_i}$

```
survey_ratio(
  numerator,
  denominator,
  na.rm = FALSE,
  vartype = c("se", "ci", "var", "cv"),
  level = 0.95,
  deff = FALSE,
  df = NULL,
  ...
)
```

`survey_ratio` Example: What is the average dollar per BTU spent on energy?

```
recs_des |>
   summarize(
      DolPerBTU=survey_ratio(
         numerator = TOTALDOL,
         denominator = TOTALBTU,
         na.rm = TRUE
         )
      )
```

```
## # A tibble: 1 × 2
##   DolPerBTU DolPerBTU_se
##       <dbl>        <dbl>
## 1    0.0241     0.000217
```

Practice time

- Open ContinuousExercises.Rmd and work through Part 1
- We will take 20 minutes. Use this time for the exercises and questions.

Weighted Analysis for Continuous Variables: Domain Analysis

- If we want to get estimates by another variable, we need to add a `group_by` statement before doing the analysis.
- Example: What is the average amount of dollars spent on electricity for households that use AC and those that do not use AC?

```
recs_des |>
  group_by(ACUsed) |>
  summarize(
    ElBill=survey_mean(DOLLAREL,
                       na.rm=TRUE)
  )
```

```
## # A tibble: 2 × 3
##   ACUsed ElBill ElBill_se
##   <lgl>   <dbl>     <dbl>
## 1 FALSE    972.      25.8
## 2 TRUE    1435.      15.8
```

Domain Analysis: Totals

- If we want the overall average electric bill too, use the `cascade` function instead of `summarize`

```
recs_des |>
   group_by(ACUsed) |>
   cascade(
      ElBill=survey_mean(DOLLAREL,
                         na.rm=TRUE)
   )
```

```
## # A tibble: 3 × 3
##   ACUsed ElBill ElBill_se
##   <lgl>   <dbl>     <dbl>
## 1 FALSE    972.      25.8
## 2 TRUE    1435.      15.8
## 3 NA      1375.      14.1
```

Note: The overall average electric bill appears as NA

Domain Analysis: Totals

- Also can add sample and pop sizes

```
recs_des |>
   group_by(ACUsed) |>
   cascade(
      ElBill=survey_mean(DOLLAREL, na.rm=TRUE),
      N=survey_total(!is.na(DOLLAREL)),
      n=unweighted(sum(!is.na(DOLLAREL)))
   )
```

```
## # A tibble: 3 × 6
##   ACUsed ElBill ElBill_se          N        N_se     n
##   <lgl>   <dbl>     <dbl>      <dbl>       <dbl> <int>
## 1 FALSE    972.      25.8  15401242. 976901.       737
## 2 TRUE    1435.      15.8 102807008. 976901.      4949
## 3 NA      1375.      14.1 118208250.      0.0320  5686
```

Weighted Analysis for Specific Subpopulations

- filtering (subsetting) the data should be done AFTER specifying the design to ensure accurate standard errors
- Use the `filter` function after creating the survey design object and before summarizing

Wrong way:

```
data |>
  filter(state=="NC") |>
  as_survey_design(...) |>
  summarize(AvgAge=mean(Age))
```

Right way:

```
data |>
  as_survey_design(...) |>
  filter(state=="NC") |>
  summarize(AvgAge=mean(Age))
```

Subpopulation Example: Average electric cost of single family homes

```
recs_des |>
  filter(HousingUnitType %in% c("Single-family detached",
                                "Single-family attached")) |>
  summarize(
    ElBill=survey_mean(DOLLAREL,
                       na.rm=TRUE)
  )
```

```
## # A tibble: 1 × 2
##   ElBill ElBill_se
##    <dbl>     <dbl>
## 1  1542.      17.2
```

Comparisons with t-tests: `svyttest` Syntax

- t-tests are done in the package `survey` not `srvyr` but you can use the same design object

```
svyttest(formula, # outcome~group for two-sample, outcome~0 for one-sample
         design,
         na.rm = FALSE
         ....)
```

`svyttest` Syntax with `|>`

```
recs_des |>
   svyttest(formula=,
            design=_,
            na.rm=TRUE)
```

`svyttest` Syntax with `|>`

```
recs_des |>
   svyttest(design=_,
            formula=,
            na.rm=TRUE)
```

`svyttest` Example 1: One-sample t-test

- I keep my house at 68 degrees at night during the summer. Is this different from the national average?

```
recs_des |>
   svyttest(design=_,
            formula=I(SummerTempNight-68)~0,
            na.rm=TRUE)
```

```
##
##     Design-based one-sample t-test
##
## data:  I(SummerTempNight - 68) ~ 0
## t = 41.013, df = 94, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  3.424776 3.773247
## sample estimates:
##     mean
## 3.599012
```

`svyttest` Example 2: Comparing two variables

- Do people keep their house the same temperature at night during the summer and the winter?

```
recs_des |>
   svyttest(design=_,
            formula=I(SummerTempNight-WinterTempNight)~0,
            na.rm=TRUE)
```

```
##
##     Design-based one-sample t-test
##
## data:  I(SummerTempNight - WinterTempNight) ~ 0
## t = 29.079, df = 94, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  2.995084 3.434072
## sample estimates:
##     mean
## 3.214578
```

`svyttest` Example 3: Two-sample t-test

- Are electric bills different between those with and without A/C?

```
recs_des |>
   svyttest(design=_,
            formula=DOLLAREL~ACUsed,
            na.rm=TRUE)
```

```
##
##     Design-based t-test
##
## data:  DOLLAREL ~ ACUsed
## t = 14.772, df = 94, p-value < 2.2e-16
## alternative hypothesis: true difference in mean is not equal to 0
## 95 percent confidence interval:
##  400.6588 525.0903
## sample estimates:
## difference in mean
##           462.8746
```

Linear Regression: `svyglm` Syntax

- As with t-tests, regressions are done in the package `survey` not `srvyr` but you can use the same design object
- Syntax is similar between t-test and glm

```
svyglm(formula,
       design,
       na.action, #default is na.omit
       ....)
```

`svyglm` Example 1: Dummy regression

Same example as two-sample t-test: Are electric bills different between those with and without A/C?

t-test:

```
recs_des |>
   svyttest(design=_,
            formula=DOLLAREL~ACUsed,
            na.rm=TRUE)
```

glm:

```
recs_des |>
   svyglm(design=_,
          formula=DOLLAREL~ACUsed,
          na.action=na.omit)
```

`svyglm` Example 1: Dummy regression

Are electric bills different between those with and without A/C?

```
recs_des |>
   svyglm(design=_,
          formula=DOLLAREL~ACUsed,
          na.action=na.omit) |>
  summary()
```

```
##
## Call:
## svyglm(design = recs_des, formula = DOLLAREL ~ ACUsed, na.action = na.omit)
##
## Survey design:
## Called via srvyr
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   972.09      25.81   37.66   <2e-16 ***
## ACUsedTRUE    462.87      31.33   14.77   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 623148.2)
##
## Number of Fisher Scoring iterations: 2
```

`svyglm` Example 2: Multiple dummies

Does temperature of AC at night vary by region?

```
recs_des |>
   svyglm(design=_,
          formula=SummerTempNight~Region,
          na.action=na.omit) |>
  summary()
```

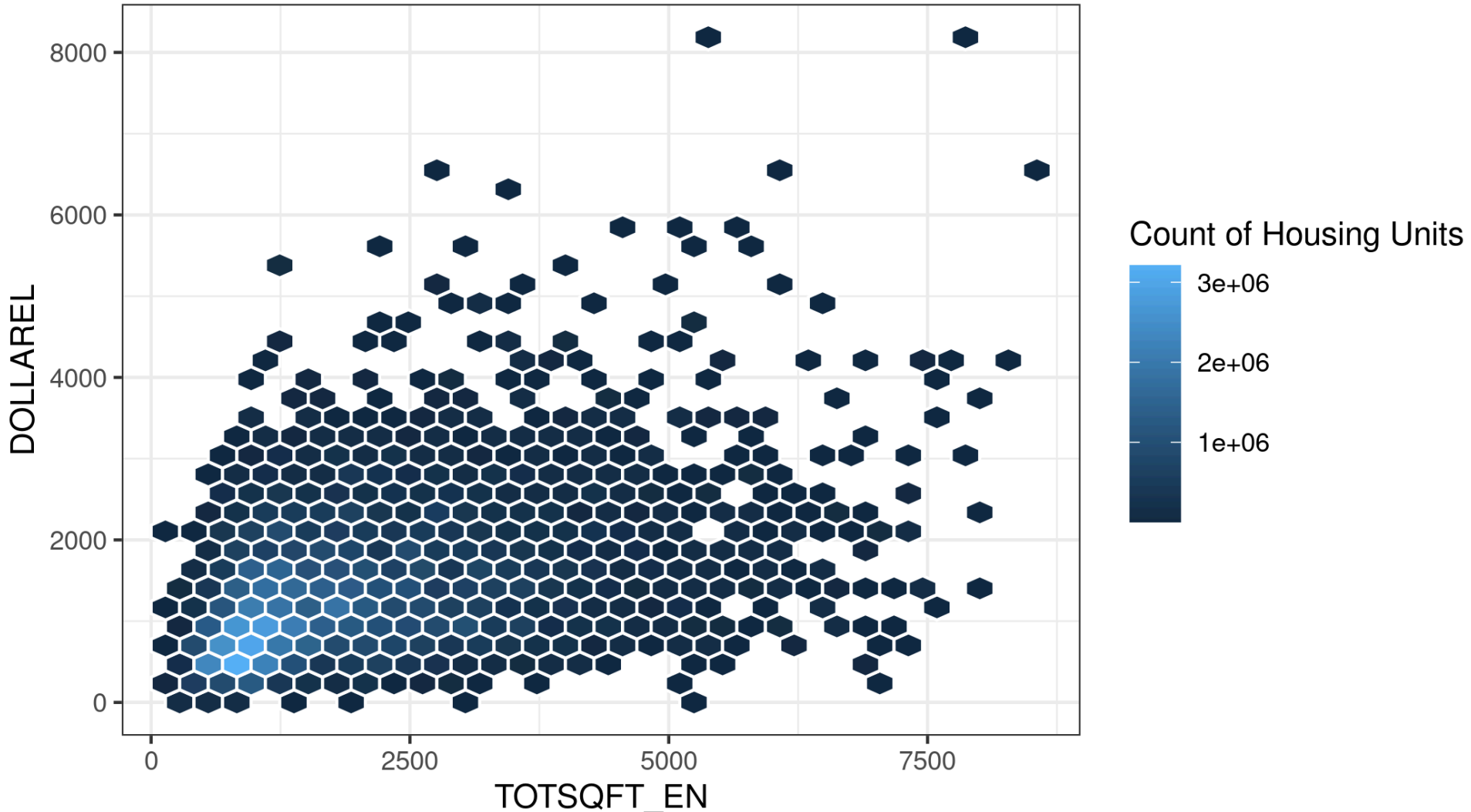
```
##
## Call:
## svyglm(design = recs_des, formula = SummerTempNight ~ Region,
##     na.action = na.omit)
##
## Survey design:
## Called via srvyr
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    70.4848     0.1968 358.151  < 2e-16 ***
## RegionMidwest   0.8744     0.2526   3.461 0.000818 ***
## RegionSouth     1.4865     0.2306   6.446 5.20e-09 ***
## RegionWest      1.6568     0.3529   4.695 9.27e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 24.06041)
##
## Number of Fisher Scoring iterations: 2
```

`svyglm` Example 3: Continuous predictor

- Is there a relationship between square footage and electric bill?
- Let's review the data first with a ggplot. *Note we use the original data and do **NOT** use the survey design object.*

```
p <- recs |>
  ggplot(aes(x=TOTSQFT_EN, y=DOLLAREL, weight=NWEIGHT)) +
  geom_hex(color="white") +
  scale_fill_gradient(guide="colourbar",name="Count of Housing Units")
```

`svyglm` Example 3: Continuous predictor



`svyglm` Example 3: Continuous predictor

```
m_electric_sqft <- recs_des |>
   svyglm(design=_,
          formula=DOLLAREL~TOTSQFT_EN,
          na.action=na.omit)
summary(m_electric_sqft)
```

```
##
## Call:
## svyglm(design = recs_des, formula = DOLLAREL ~ TOTSQFT_EN, na.action = na.omit)
##
## Survey design:
## Called via srvyr
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 879.89542   26.31370   33.44   <2e-16 ***
## TOTSQFT_EN    0.24633    0.01338   18.42   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 549674.3)
##
## Number of Fisher Scoring iterations: 2
```

Practice time

- Open ContinuousExercises.Rmd and work through Part 2
- We will take 20 minutes. Use this time for the exercises and questions.

Categorical Dependent Variables

Weighted Analysis for Categorical Variable

- Functions to use within `summarize` after `group_by`
    - survey_mean/survey_prop
    - survey_total
- Functions to get counts
    - survey_count

`survey_count` Syntax

- `survey_count` functions similarly to `count` in that it is **NOT** called within `summarize`
- Produces weighted counts and variance of your choice of those counts

```
survey_count(
   x,
   ...,
   wt = NULL,
   sort = FALSE,
   name = "n",
   .drop = dplyr::group_by_drop_default(x),
   vartype = c("se", "ci", "var", "cv")
)
```

`survey_count` Example

- Cross-tab of population in each age group and gender

```
anes_des |>
   survey_count(AgeGroup, Gender, name="N")
```

```
## # A tibble: 21 × 4
##    AgeGroup Gender         N     N_se
##    <fct>    <fct>      <dbl>    <dbl>
##  1 18-29    Male   21600792. 1418333.
##  2 18-29    Female 22193812. 1766188.
##  3 18-29    <NA>      65204.   56033.
##  4 30-39    Male   19848178. 1077514.
##  5 30-39    Female 19780778. 1158766.
##  6 30-39    <NA>     118195.   62999.
##  7 40-49    Male   17915676. 1123493.
##  8 40-49    Female 18932548.  946369.
##  9 40-49    <NA>      71911.   55174.
## 10 50-59    Male   19054298. 1029844.
## # ℹ 11 more rows
```

`survey_mean` and `survey_total` within `summarize`

- Specify the sample design,
- then specify the crosstab in `group_by`,
- then `survey_mean` or `survey_prop` used with no x (variable) calculates a proportion of groups within `summarize`, or
- `survey_total` used with no x (variable) calculates a population count estimate within `summarize`

`survey_mean` and `survey_prop` Syntax

```
survey_mean(
   x,
   na.rm = FALSE,
   vartype = c("se", "ci", "var", "cv"),
   level = 0.95,
   proportion = FALSE,
   prop_method = c("logit", "likelihood", "asin", "beta", "mean"),
   deff = FALSE,
   df = NULL,
   ...
)

survey_prop(
  vartype = c("se", "ci", "var", "cv"),
  level = 0.95,
  proportion = FALSE,
  prop_method = c("logit", "likelihood", "asin", "beta", "mean"),
  deff = FALSE,
  df = NULL,
  ...
)
```

`survey_mean` and `survey_total` Examples

Looking at population by age group as done with `survey_count`.

```
anes_des |>
   group_by(AgeGroup) |>
   summarize(
      p1=survey_mean(),
      p2=survey_prop(),
      N=survey_total(),
      n=unweighted(n()), # this gets unweighted counts aka sample sizes
      .groups="drop" # summarize option to remove groups
   )
```

```
## # A tibble: 7 × 8
##   AgeGroup        p1   p1_se     p2   p2_se         N     N_se     n
##   <fct>        <dbl>   <dbl>  <dbl>   <dbl>     <dbl>    <dbl> <int>
## 1 18-29       0.189  0.00838 0.189  0.00838 43859809. 2340503.   871
## 2 30-39       0.172  0.00659 0.172  0.00659 39747151. 1556193.  1241
## 3 40-49       0.159  0.00609 0.159  0.00609 36920134. 1452300.  1081
## 4 50-59       0.169  0.00657 0.169  0.00657 39191266. 1602082.  1200
## 5 60-69       0.155  0.00488 0.155  0.00488 35833416. 1214320.  1436
## 6 70 or older 0.119  0.00474 0.119  0.00474 27503517. 1146535.  1330
## 7 <NA>        0.0369 0.00305 0.0369 0.00305  8537401.  710907.   294
```

Conditional proportions with more than one group

- Specifying more than one group calculates conditional proportions
- Example: people voting in 2016 and 2020

```
anes_des |>
   filter(!is.na(VotedPres2016), !is.na(VotedPres2020)) |>
   group_by(VotedPres2016, VotedPres2020) |>
   summarize(
      p=survey_mean(),
      N=survey_total(),
      n=unweighted(n()),
      .groups="drop"
   )
```

```
## # A tibble: 4 × 7
##   VotedPres2016 VotedPres2020      p    p_se          N     N_se     n
##   <fct>         <fct>          <dbl>   <dbl>      <dbl>    <dbl> <int>
## 1 Yes           Yes           0.924  0.00566 144578247. 2617349.  5534
## 2 Yes           No            0.0762 0.00566  11917394.  955174.   274
## 3 No            Yes           0.455  0.0162   33923120. 1594478.   859
## 4 No            No            0.545  0.0162   40606907. 2036095.   761
```

Joint proportions with more than one group

- Specify an interaction to get joint distribution - use `interact` within `group_by`
- Example: people voting in 2016 and 2020

```
anes_des |>
   filter(!is.na(VotedPres2020), !is.na(VotedPres2016)) |>
   group_by(interact(VotedPres2016, VotedPres2020)) |>
   summarize(
      p=survey_mean(),
      N=survey_total(),
      .groups="drop"
   )
```

```
## # A tibble: 4 × 6
##   VotedPres2016 VotedPres2020      p    p_se          N     N_se
##   <fct>         <fct>          <dbl>   <dbl>      <dbl>    <dbl>
## 1 Yes           Yes            0.626  0.00934 144578247. 2617349.
## 2 Yes           No             0.0516 0.00391  11917394.  955174.
## 3 No            Yes            0.147  0.00628  33923120. 1594478.
## 4 No            No             0.176  0.00770  40606907. 2036095.
```

Proportions with Design Effects

```
anes_des |>
   filter(!is.na(VotedPres2016), !is.na(VotedPres2020)) |>
   group_by(interact(VotedPres2016, VotedPres2020)) |>
   summarize(
      p=survey_mean(deff=TRUE),
      N=survey_total()
   )
```

```
## # A tibble: 4 × 7
##   VotedPres2016 VotedPres2020      p    p_se p_deff          N     N_se
##   <fct>         <fct>          <dbl>   <dbl>  <dbl>      <dbl>    <dbl>
## 1 Yes           Yes           0.626  0.00934   2.76 144578247. 2617349.
## 2 Yes           No            0.0516 0.00391   2.32  11917394.  955174.
## 3 No            Yes           0.147  0.00628   2.34  33923120. 1594478.
## 4 No            No            0.176  0.00770   3.04  40606907. 2036095.
```

Proportions: confidence intervals

```
anes_des |>
   group_by(interact(Income7, VotedPres2016, VotedPres2020)) |>
   summarize(
      pd=survey_prop(vartype="ci") |> round(4)
   ) |> select(Income7, VotedPres2016, VotedPres2020, pd, contains("_")) |>
   DT::datatable(fillContainer = FALSE, options = list(pageLength = 4))
```

Proportions: confidence intervals (results)

Show 4 entries Search:

	Income7	VotedPres2016	VotedPres2020	pd	pd_low	pd_upp
1	Under $20k	Yes	Yes	0.0332	0.0289	0.038
2	Under $20k	Yes	No	0.0064	0.0045	0.0091
3	Under $20k	No	Yes	0.0141	0.0113	0.0177
4	Under $20k	No	No	0.0327	0.0269	0.0397

Showing 1 to 4 of 45 entries Previous 1 2 3 4 5 ... 12 Next

Logistic regression with **svyglm**

```
svyglm(formula, # response ~ terms
       design,
       na.action, #default is na.omit
       family = quasibinomial, # use this to avoid warning about non-integers
       ....)
```

Example logistic regression

- Predicting trust in government by who someone voted in 2020

```
filter(anes_des, Weight>0) |>
   svyglm(design=_,
          formula=TrustGovernment~ VotedPres2020_selection,
          family = quasibinomial) |>
   summary()
```

```
##
## Call:
## svyglm(formula = TrustGovernment ~ VotedPres2020_selection, design = filter(anes_des,
##     Weight > 0), family = quasibinomial)
##
## Survey design:
## Called via srvyr
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    4.6785     0.3266  14.323   <2e-16 ***
## VotedPres2020_selectionTrump  -0.3530     0.4008  -0.881   0.3829
## VotedPres2020_selectionOther   2.5265     1.0868   2.325   0.0243 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Practice time

- Open CategoricalExercises.Rmd and work through the questions
- We will take 30 minutes. Use this time for the exercises and questions.

Closing

Resources for more learning

- Zimmer, Powell, and Velásquez. 2024. Exploring Complex Survey Data Analysis Using R.
- Online documentation for the sampling package
- https://cran.r-project.org/web/packages/srvyr/vignettes/srvyr-vs-survey.html
- https://r-survey.r-forge.r-project.org/survey/ (Includes more advanced modeling)

Data Sources

- The American National Election Studies (https://electionstudies.org/). These materials are based on work supported by the National Science Foundation under grant numbers SES 1444721, 2014-2017, the University of Michigan, and Stanford University.
- *Residential Energy Consumption Survey: Using the 2015 Microdata File to Compute Estimates and Standard Errors.* U.S. Department of Energy (2017) https://www.eia.gov/consumption/residential/data/2015/pdf/microdata_v3.pdf