Regression modeling with complex survey data

Week 10 (ch 11)

Stat 260, St. Clair

Goal:

- A regression model describes how a response y varies as a function of explanatory variables x
- Typical regression modeling goals:

1. Describe the relationship between variables.
2. Predict a response y given x
3. Determine how changes in x **cause** changes in y

Model-based regression: Stat 230

- Build a theoretical "universal" model for y given x that holds across populations
- Describe a "data generating model" (DGM)
    - a stochastic model that “generates” the particular finite population of individuals
- A model comes with structural probabilistic assumptions that must be checked

Model-based regression: Stat 230

- Variables:
    - Response Y
    - Covariates (predictors/explanatory) x
- Simple linear regression model: describes the **conditional probability distribution of y given x**

$$Y_i \mid x_i \sim N(\mu_i, \sigma^2) \quad \mu_i = \beta_0 + \beta_1 x_i$$

- Model assumptions:

(1) Linear relationship

(2) Constant variance

(3) Normally distributed

(4) Independence

Model-based regression: estimation

- Obtain data we believe was generated by a particular DGP
- Use **maximum likelihood** inference methods to derive parameter estimates and SE for theoretical parameters β_0, β_1, and σ
  - only based on the model assumptions, not sampling weights!
- e.g. the slope estimate:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n x_i y_i - \frac{1}{n} \sum_{i=1}^n x_i \sum_{i=1}^n y_i}{\sum_{i=1}^n x_i^2 - \frac{1}{n} \left(\sum_{i=1}^n x_i\right)^2}$$

- But estimates and their SE's are highly dependent upon model assumptions (1), (2) and (4)

Design-based regression: Stat 260

- Population parameters B_0 and B_1 are the "best fit" intercept and slope for the population trend

$$y = B_0 + B_1 x$$

- "best fit" means B_0 and B_1 minimize

$$\sum_{i=1}^N (y_i - B_0 - B_1 x_i)^2$$

- e.g. the population slope is

$$B_1 = \frac{\sum_{i=1}^N x_i y_i - \frac{1}{N} \sum_{i=1}^N x_i \sum_{i=1}^N y_i}{\sum_{i=1}^N x_i^2 - \frac{1}{N} \left(\sum_{i=1}^N x_i\right)^2}$$

Design-based regression: estimation

- B_1 is just another population parameter to estimate using sampling weights
  - Model fit is not important since there is no model structure!
- e.g B_1 is just a function of population totals so we use an appropriately weighted estimate:

$$\hat{B}_1 = \frac{\sum_{i=1}^n w_i x_i y_i - \frac{1}{\sum_{i=1}^n w_i} \sum_{i=1}^n w_i x_i \sum_{i=1}^n w_i y_i}{\sum_{i=1}^n w_i x_i^2 - \frac{1}{\sum_{i=1}^n w_i} \left(\sum_{i=1}^n w_i x_i\right)^2}$$

- Shouldn't apply design-based parameter estimates $\hat{B}_0, \hat{B}_1$ to other finite populations.

Design-based vs model-based regression

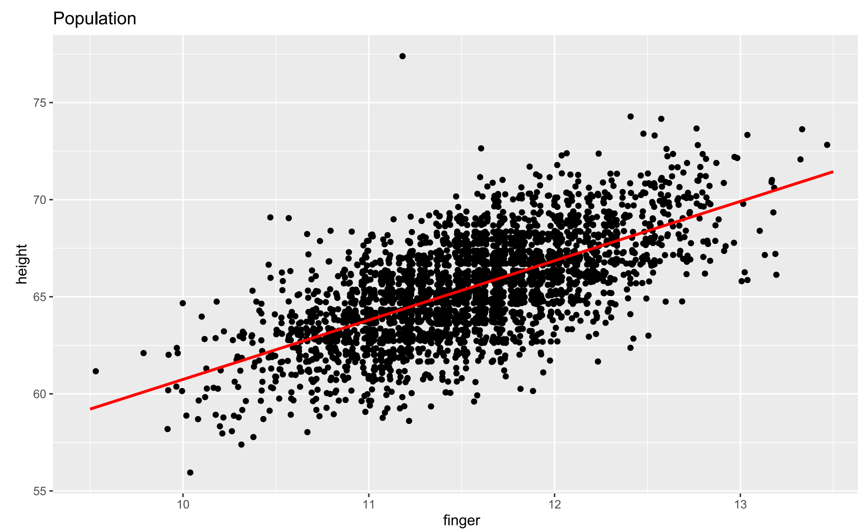
- Can think of the finite population of y_i's as being a realization from a "universal" DGM described earlier
  - then B's should be close to β's
- If **estimates** of B_1 and β_1 differ by a lot, then this could indicate that the **model** is inadequate
  - the model doesn't fit all subpopulations well
  - sampling weights are likely accounting for some unmeasured variable that is important to the relationship between y and x
- Models can include design variables
  - use stratification variables as covariates
  - fit a mixed-effects model with random cluster effects (Stat 330)

Example: The population

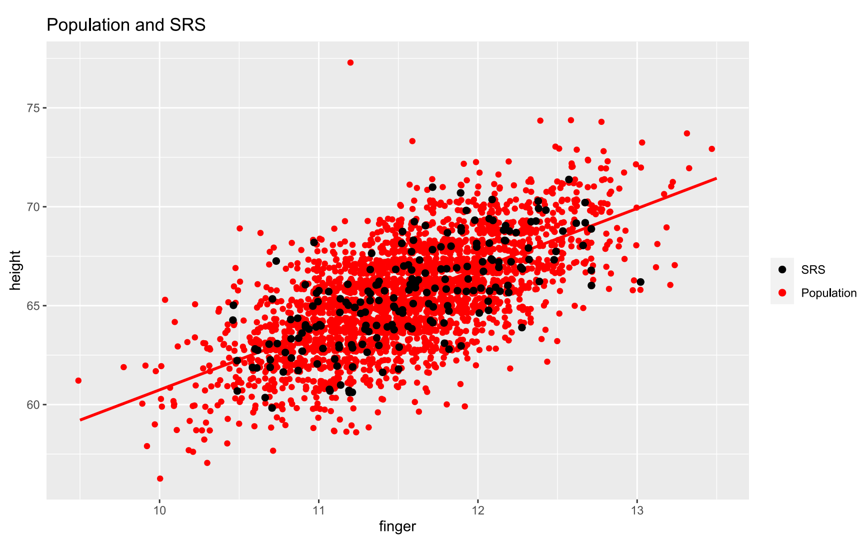
- `anthrop` in `SDaA`
  - A population of 3000 late 19th century "criminals" (anthrop.csv)
- Goal: model height as a function of finger length

```
pop <- anthrop  # the finite pop.
str(pop)
## tibble [3,000 x 2] (S3: tbl_df/tbl/data.frame)
##  $ finger: num [1:3000] 10 10.3 9.9 10.2 10.2 10.3 10.4 10.7 10 1
##  $ height: num [1:3000] 56 57 58 58 58 58 58 58 59 59 ...
##  - attr(*, "label")= chr "ANTHROP                              "
pop_lm <- lm(height ~ finger, data=pop)
pop_lm
##
## Call:
## lm(formula = height ~ finger, data = pop)
##
## Coefficients:
## (Intercept)       finger
##      30.179        3.056
```


Example: The population



Example: The SRS of size 200 `anthsrs`



Example: The SRS of size 200 `anthsrs`

- With an SRS, the model- and design-based estimates are the same (self-weighting).
- Model-based estimation:

```
anthsrs_lm<- lm(height~finger, data= anthsrs)   # model-based
broom::tidy(anthsrs_lm)
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     30.3      2.57      11.8 1.03e-24
## 2 finger           3.05     0.222     13.7 1.36e-30
```

Example: The SRS of size 200 `anthsrs`

- Design-based estimation:

```
anthsrs$N<- 3000
anthsrs$wts<- 3000/200
anthsrs_design<- svydesign(id = ~1,
                           fpc = ~N,
                           weights = ~wts,
                           data = anthsrs)
anthsrs_svylm<- svyglm(height ~ finger,
                       design = anthsrs_design)
broom::tidy(anthsrs_svylm)
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    30.3      2.46       12.3 2.60e-26
## 2 finger          3.05     0.213      14.3 2.12e-32
```


Example: The SRS of size 200 `anthsrs`

- Finite population:

$$B_1 = 3.056, \quad B_0 = 30.179$$

- Model-based slope estimate:

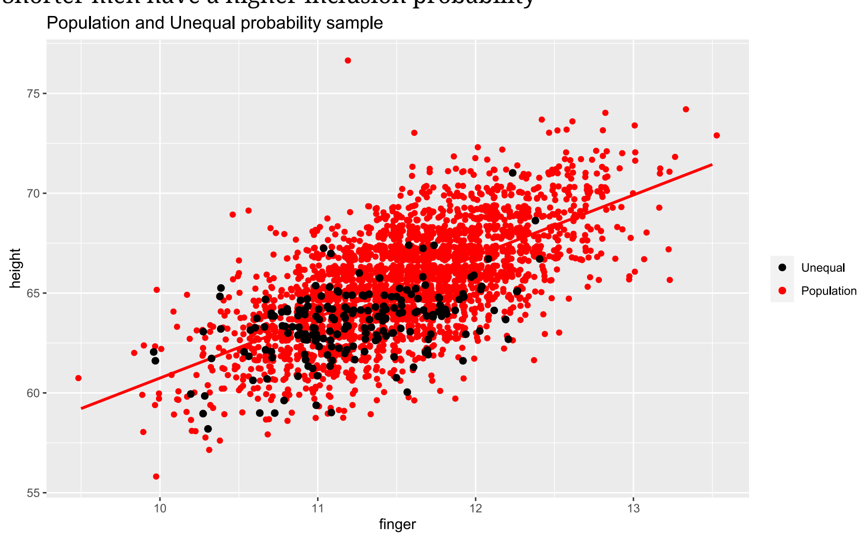
$$\hat{\beta}_1 = 3.0453(SE = 0.2217), \quad \hat{\beta}_0 = 30.3162(SE = 2.5668)$$

- Design-based slope estimate:

$$\hat{B}_1 = 3.0453(SE = 0.2126), \quad \hat{B}_0 = 30.3162(SE = 2.4574)$$

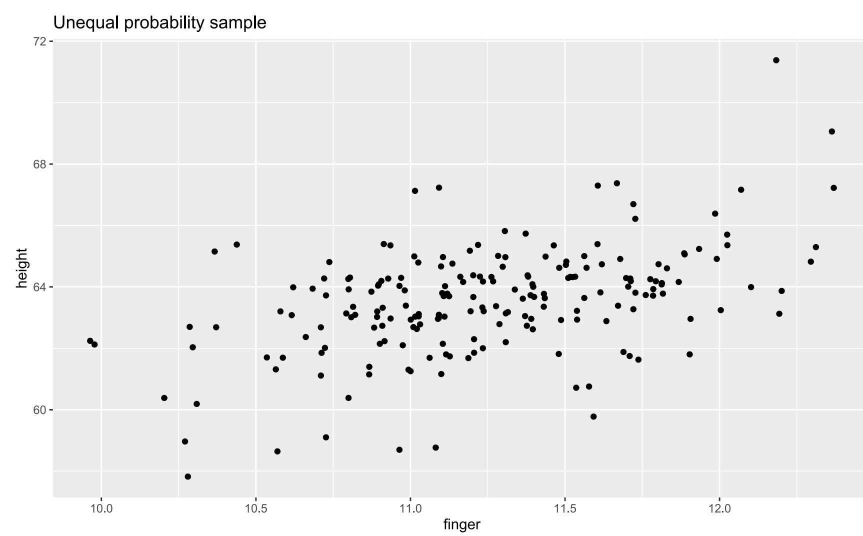
Example: unequal probability sample `anthuneq`

Shorter men have a higher inclusion probability



Example: unequal probability sample `anthuneq`

But we can't see the fact that shorter men are overrepresented in the usual data scatterplot



Example: unequal probability sample `anthuneq`

- `svyplot`: circle size is proportional to sampling weight

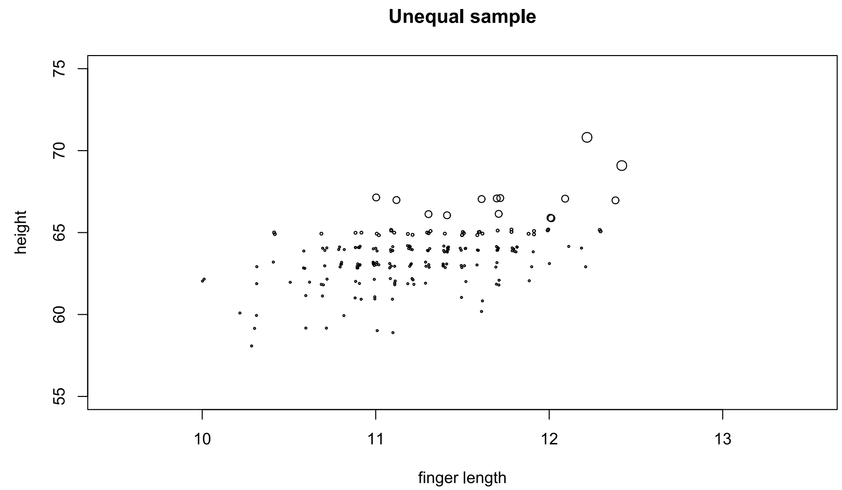
```
anthuneq_design <- svydesign(id=~1, weight = ~wt, data= anthuneq)
svyplot(jitter(height) ~ jitter(finger),
        design = anthuneq_design)
```

- `svyplot`: `style="hex"` uses hexagonal binning that sums weights by bin groups
  - may need to install `hexbin` package

```
svyplot(jitter(height) ~ jitter(finger),
        design = anthuneq_design,
        style = "hex")
```

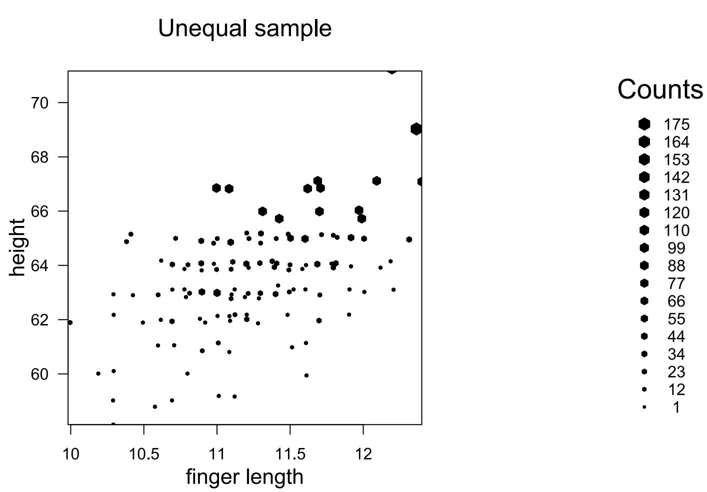

Example: unequal probability sample `anthuneq`

`svyplot`: circle size is proportional to sampling weight



Example: unequal probability sample `anthuneq`

`svyplot`: `hex` style (visually better for larger data sets)



Example: unequal probability sample `anthuneq`

- Model-based estimation:

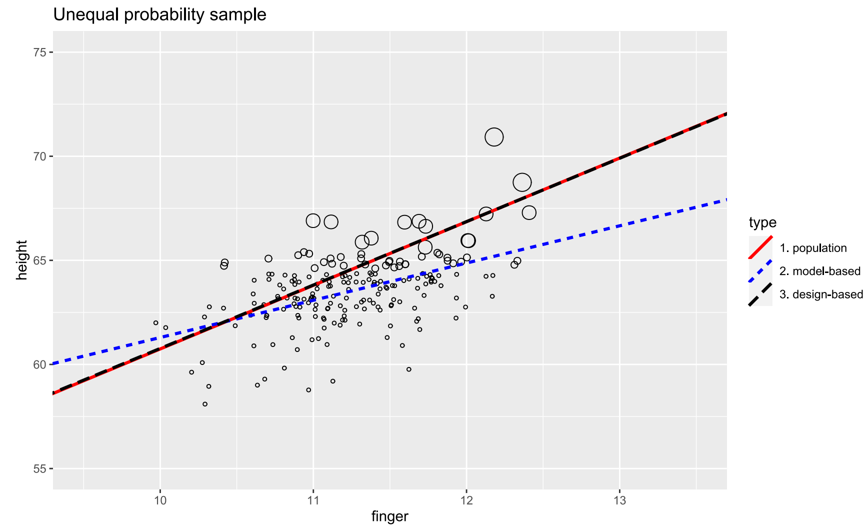
```
anthuneq_lm <- lm(height ~ finger, data = anthuneq)   # model-based
broom::tidy(anthuneq_lm)
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    43.4      2.55      17.0  1.15e-40
## 2 finger          1.79     0.226      7.90 1.87e-13
```

Example: unequal probability sample `anthuneq`

- Design-based estimation:

```
anthuneq_svylm <- svyglm(height ~ finger, design=anthuneq_design)
broom::tidy(anthuneq_svylm)
## # A tibble: 2 x 5
##   term        estimate std.error statistic      p.value
##   <chr>          <dbl>     <dbl>     <dbl>        <dbl>
## 1 (Intercept)    30.2      6.63       4.56 0.00000913
## 2 finger          3.05     0.588      5.19 0.000000512
```


Example: unequal probability sample `anthuneq`



Example: unequal probability sample `anthuneq`

- Finite population:

$$B_1 = 3.056, \quad B_0 = 30.179$$

- Model-based slope estimate:

$$\hat{\beta}_1 = 1.7886(SE = 0.2263), \quad \hat{\beta}_0 = 43.4079(SE = 2.5481)$$

- Design-based slope estimate:

$$\hat{B}_1 = 3.0550(SE = 0.5883), \quad \hat{B}_0 = 30.1753(SE = 6.6284)$$

- Inference about the *population* of all criminals is not estimated correctly by the model-based solution!

`ggplot2` options

- Add `size` aesthetic to make circle size is proportional to sampling weight
- Add `weight` aesthetic to `geom_smooth` to add the weighted (design-based) regression line

```
ggplot(anthuneq, aes(x = finger, y = height)) +
  geom_jitter(aes(size = wt), shape = 1, show.legend = FALSE)  +
  geom_smooth(aes(weight = wt), method = "lm", se = FALSE)
```