# 2022 SISCER SAE: The `survey` package

Jon Wakefield and Peter Gao
Departments of Statistics and Biostatistics
University of Washington

2022-07-07

### `R` Survey Package

Written by Thomas Lumley, who has a site for the package here:

http://r-survey.r-forge.r-project.org/survey/

The site for the book Lumley (2010, Complex Surveys: A Guide to Analysis using R) is here:

http://r-survey.r-forge.r-project.org/svybook/index.html

### Sampling in `R`

For simple random sampling we require $n$ from $N$.

Suppose we have a list of the $N$ units from which we wish to sample $n$, the `sample` command allows SRS without replacement.

```
N <- 5000
n <- 12
sample(N, n, replace = FALSE)
##  [1] 1352 4600 1933 3004 2739 3965 2801 2177 4122 2245 3460 4190
```

This mimics an SRS sampling design of $n = 12$ from a population of $N = 5000$. The numbers produced are the indices of the units of the population that were sampled.

### Academic Performance Index (API) data

- This is a useful pedagogic dataset because the complete population data are available, along with various probability samples.
- The schools are the observation units.
- From the help file (Type "`?api`" at the console): The Academic Performance Index is computed for all California schools based on standardised testing of students and other factors such as attendance and graduation rates.
- A numeric API score ranges from a low of 200 to a high of 1000.
- The data sets contain information for all schools with at least 100 students.
- Full data: 6194 observations (schools) on the 37 variables including `cds` (unique identifier), `stype` (Elementary/Middle/High), `api00` (API in 2000),...

From `?api`:

- `apipop` is the entire population, apisrs is a simple random sample, `apiclus1` is a cluster sample of school districts, `apistrat` is a sample stratified by `stype` (school type), and `apiclus2` is a two-stage cluster sample of schools within districts.

Each row of the data contains data on one school, i.e., contains information on the children within that school.

### API data: non-survey package R commands

We first look at the SRS dataset since for this we can use regular (non-survey) R methods since we have a random sample (i.e., no weighting required).

In this sample, 200 schools are randomly sampled and 39 variables are available.

The survey package needs to be loaded since it contains the API data.

```
library(survey)
data(api)
names(apisrs)
##  [1] "cds"      "stype"    "name"     "sname"    "snum"     "dname"
##  [7] "dnum"     "cname"    "cnum"     "flag"     "pcttest"  "api00"
## [13] "api99"    "target"   "growth"   "sch.wide" "comp.imp" "both"
## [19] "awards"   "meals"    "ell"      "yr.rnd"   "mobility" "acs.k3"
## [25] "acs.46"   "acs.core" "pct.resp" "not.hsg"  "hsg"      "some.col"
## [31] "col.grad" "grad.sch" "avg.ed"   "full"     "emer"     "enroll"
## [37] "api.stu"  "pw"       "fpc"
```

We find out the size of the dataframe apisrs, and then evaluate the mean and variance of the enroll variable (over the sample of schools), which is the number of children enrolled in the school.

```
dim(apisrs)
## [1] 200  39
ybar <- mean(apisrs$enroll)
sd <- sqrt(var(apisrs$enroll))
ybar
## [1] 584.61
sd
## [1] 393.4514
```

Note the standard error is for the infinite population mean parameter $\mu = E[Y]$, not the finite sample mean $\overline{Y}_U$, since no finite population correction factor.

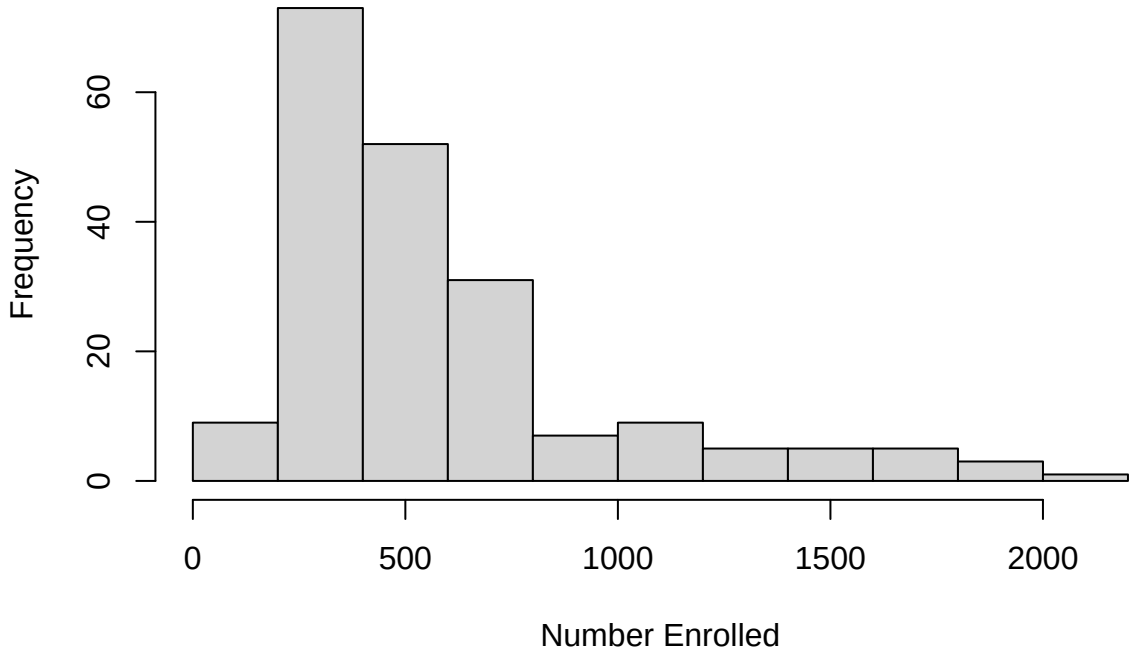
We now calcualte a 95% confidence interval for the superpopulation mean.

```
se <- sd/sqrt(length(apisrs$enroll))
endSRS <- qnorm(0.975) * se
c(ybar - endSRS, ybar + endSRS)
## [1] 530.0814 639.1386
```

### API data: Some plots

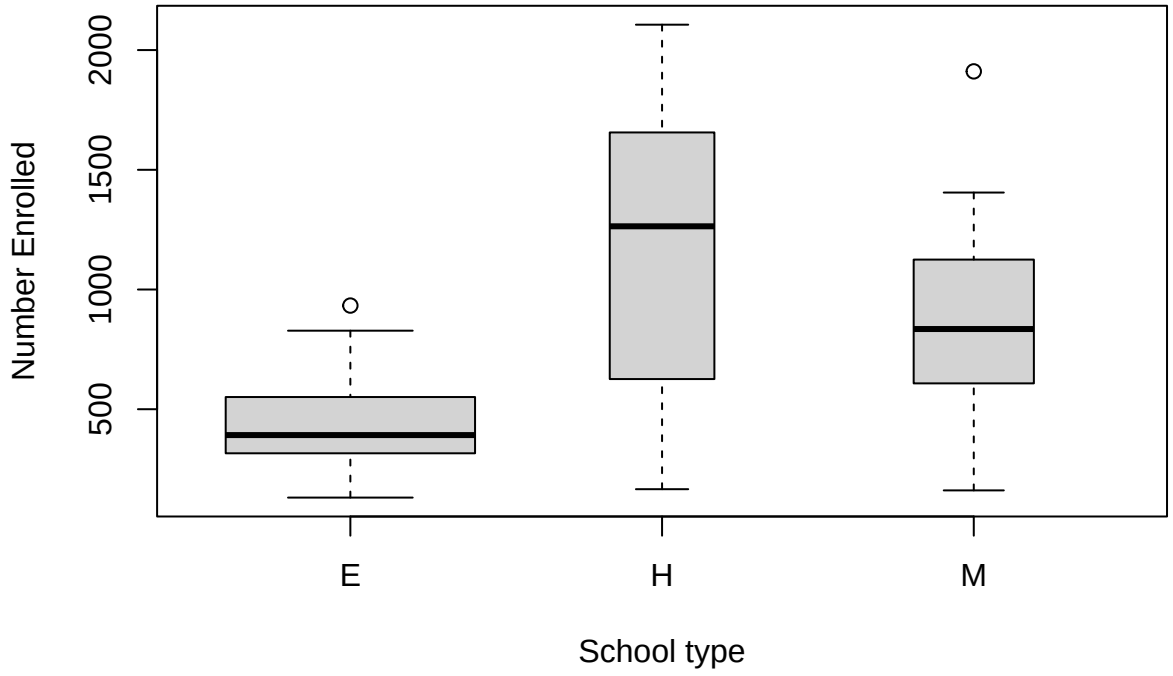
Let's look at a histogram of the number enrolled.

```
hist(apisrs$enroll, xlab = "Number Enrolled", main = "")
```

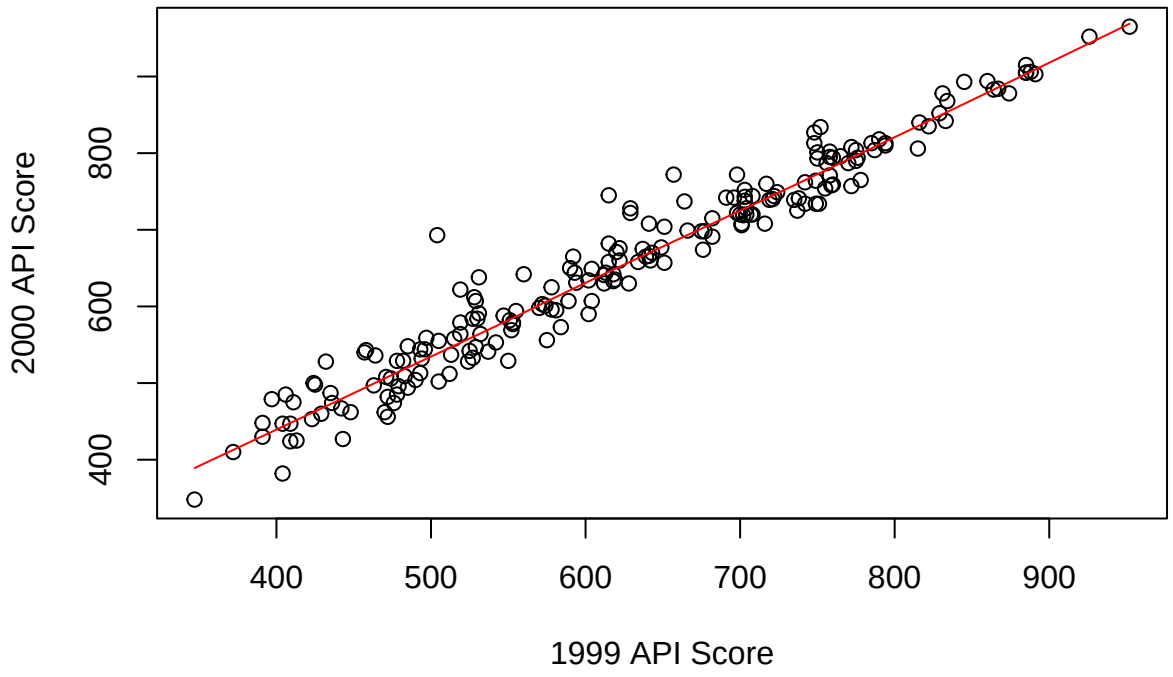



Boxplots of number enrolled by school type (elementary/high/middle).

```
boxplot(apisrs$enroll ~ apisrs$stype, xlab = "School type", ylab = "Number Enrolled",
    main = "", varwidth = TRUE)
```




```
plot(api00 ~ api99, ylab = "2000 API Score", xlab = "1999 API Score", data = apisrs)
lines(lowess(apisrs$api00 ~ apisrs$api99), col = "red")
```




## API data: Fitting a linear model

Regression coefficents and var/cov matrix of estimators.

```
linmod <- lm(api00 ~ api99, data = apisrs)
coef(linmod)
## (Intercept)       api99
##  63.2830726   0.9497618
vcov(linmod)
##              (Intercept)         api99
## (Intercept)  94.7567807 -0.1448003030
## api99        -0.1448003  0.0002317973
```

95% CIs for intercept and slope using t distribution for endpoints

```
confint(linmod)
##                 2.5 %     97.5 %
## (Intercept) 44.086844 82.4793010
## api99        0.919738  0.9797855
```

## Defining a `survey` object

So far we have been looking at survey data gathered from a SRS, so there has been no need to do anything different to what we would do "usually".

First step in analyzing non-SRS survey data within `R` is to define a `survey` object using the `svydesign` function. From the help file (edited):

`svydesign(ids, probs=NULL, strata = NULL, variables = NULL, fpc=NULL, data = NULL,...)`

- `ids`: formula or data frame specifying cluster ids from largest level to smallest level, ~0 or ~1 is a formula for no clusters.

- `probs`: formula or data frame specifying cluster sampling probabilities.
- `strata`: formula or vector specifying strata, use NULL for no strata.
- `variables`: formula or data frame specifying the variables measured in the survey. If NULL, the data argument is use.
- `fpc`: finite population correction (may be population size or fraction of population).
- `weights`: formula or vector specifying sampling weights as an alternative to probability.
- `data`: data frame to look up variables in the formula arguments.

We first illustrate using the simple random sample version of the data `apisrs`:

```
srs_design <- svydesign(id = ~1, fpc = ~fpc, data = apisrs)
table(apisrs$fpc)
##
## 6194
##  200
```

- The argument `id=~1` says that individual schools were sampled (there is one row for each school in the data set).
- The `data=` argument specifies where the data are found.
- The argument `fpc=~fpc` says that the variable called `fpc` in the dataset contains the population size for each stratum - there is just one stratum here and so the total population size is a single number, the number of schools (these are the units).

### SRS example

Let's look at the survey object for the SRS data a little more:

```
srs_design
## Independent Sampling design
## svydesign(id = ~1, fpc = ~fpc, data = apisrs)
names(srs_design)
## [1] "cluster"    "strata"     "has.strata" "prob"       "allprob"
## [6] "call"       "variables"  "fpc"        "pps"
names(srs_design$variables)
##  [1] "cds"      "stype"    "name"     "sname"    "snum"     "dname"
##  [7] "dnum"     "cname"    "cnum"     "flag"     "pcttest"  "api00"
## [13] "api99"    "target"   "growth"   "sch.wide" "comp.imp" "both"
## [19] "awards"   "meals"    "ell"      "yr.rnd"   "mobility" "acs.k3"
## [25] "acs.46"   "acs.core" "pct.resp" "not.hsg"  "hsg"      "some.col"
## [31] "col.grad" "grad.sch" "avg.ed"   "full"     "emer"     "enroll"
## [37] "api.stu"  "pw"       "fpc"
```

```
summary(srs_design)
## Independent Sampling design
## svydesign(id = ~1, fpc = ~fpc, data = apisrs)
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.03229 0.03229 0.03229 0.03229 0.03229 0.03229
## Population size (PSUs): 6194
## Data variables:
##  [1] "cds"      "stype"    "name"     "sname"    "snum"     "dname"
##  [7] "dnum"     "cname"    "cnum"     "flag"     "pcttest"  "api00"
## [13] "api99"    "target"   "growth"   "sch.wide" "comp.imp" "both"
```

```
## [19] "awards"   "meals"    "ell"      "yr.rnd"   "mobility" "acs.k3"
## [25] "acs.46"   "acs.core" "pct.resp" "not.hsg"  "hsg"      "some.col"
## [31] "col.grad" "grad.sch" "avg.ed"   "full"     "emer"     "enroll"
## [37] "api.stu"  "pw"       "fpc"
```

```
head(srs_design$variables, n = 2)
##                  cds stype            name                        sname snum
## 1039 15739081534155     H  McFarland High                McFarland High 1039
## 1124 19642126066716     E Stowers (Cecil  Stowers (Cecil B.) Elementary 1124
##                   dname dnum       cname cnum flag pcttest api00 api99 target
## 1039 McFarland Unified  432        Kern   14   NA      98   462   448     18
## 1124       ABC Unified    1 Los Angeles   18   NA     100   878   831     NA
##      growth sch.wide comp.imp both awards meals ell yr.rnd mobility acs.k3
## 1039     14       No      Yes   No     No    44  31   <NA>        6     NA
## 1124     47      Yes      Yes  Yes    Yes     8  25   <NA>       15     19
##      acs.46 acs.core pct.resp not.hsg hsg some.col col.grad grad.sch avg.ed
## 1039     NA       24       82      44  34       12        7        3   1.91
## 1124     30       NA       97       4  10       23       43       21   3.66
##      full emer enroll api.stu    pw  fpc
## 1039   71   35    477     429 30.97 6194
## 1124   90   10    478     420 30.97 6194
```

We construct the (finite population) mean estimate and it's associated standard error "by hand" and compare with the output of `svymean`. The fpc is close to 1 here, so standard error without fpc adjustment is only slightly bigger.

```
mean(apisrs$enroll)
## [1] 584.61
fpcfact <- 1 - 200/6194
fpcfact
## [1] 0.9677107
sqrt(var(apisrs$enroll)/200) * sqrt(fpcfact)
## [1] 27.36837
SRSmean <- svymean(~enroll, srs_design)
SRSmean
##          mean     SE
## enroll 584.61 27.368
coef(lm(apisrs$enroll ~ 1))
## (Intercept)
##      584.61
sqrt(vcov(lm(apisrs$enroll ~ 1)))
##             (Intercept)
## (Intercept)    27.82121
```

We form a 95% asymptotic confidence interval for the finite population mean (not the superpopulation mean, that would not contain the fpc component).

```
confint(SRSmean)
##          2.5 %  97.5 %
## enroll 530.969 638.251
# We know the truth here!
mean(apipop$enroll, na.rm = T)
## [1] 619.0469
```

## Stratified Simple Random Sampling

### Defining a stratified survey object

We now play with the stratified random sample version of the data, `apistrat`.

The sample is stratified on school type with 100 elementary, 50 middle and 50 high schools being sampled.

```
strat_design <- svydesign(id = ~1, strata = ~stype, fpc = ~fpc, data = apistrat)
strat_design
## Stratified Independent Sampling design
## svydesign(id = ~1, strata = ~stype, fpc = ~fpc, data = apistrat)
```

- The argument `id=~1` says that individual schools were sampled (there is one row for each school in the data set).
- The `strata=~stype` gives the stratum variable, which is school type.
- The argument `weights=~pw` gives the name of the variable defining the sampling weights (we do not need to specify this variable as the population size and sample size are available).
- The argument `fpc=~fpc` says that the variable called `fpc` in the dataset contains the population size for each stratum.

We do not need both `weights` and `fpc`.

If both are left off then equal sampling probabilities are assumed (which would be incorrect here).

Weights: 4421/100=44.2, 1018/50=20.36, 755/50=15.1 for E, M, H school reflecting the oversampling/undersampling of high/elementary schools, relative to middle schools.

```
table(apistrat$pw)
##
## 15.1000003814697 20.3600006103516 44.2099990844727
##               50                50               100
table(apistrat$fpc)
##
##  755 1018 4421
##   50   50  100
table(apistrat$stype)
##
##   E   H   M
## 100  50  50
svytable(~pw, design = strat_design)
## pw
## 15.1000003814697 20.3600006103516 44.2099990844727
##              755             1018             4421
```

Stratification on a variable that is associated with an outcome of interest, can dramatically increase the efficiency of estimation - we are leveraging the association, and the known stratum totals.

Estimation is carried out using a weighted estimator, and the standard error is calculated using the appropriate (design-based) formula that accounts for the stratification.

Stratification reduces the standard error, as compared to SRS, by about 1/3.

```
STRATmean <- svymean(~enroll, strat_design)
STRATmean
##           mean     SE
## enroll 595.28 18.509
sqrt(vcov(STRATmean)/vcov(SRSmean))
```

```
##           enroll
## enroll 0.6762739
```

We can also estimate the total enrollment

```
svytotal(~enroll, strat_design)
##          total     SE
## enroll 3687178 114642
# Truth
sum(apipop$enroll, na.rm = T)
## [1] 3811472
```

## Stratified random sample

There is no uncertainty in the estimated number of schools of each type with the stratified sample, because we know the type of school for all members of the population

```
svytotal(~stype, strat_design)
##        total SE
## stypeE  4421  0
## stypeH   755  0
## stypeM  1018  0
```

Compare with the SRS design:

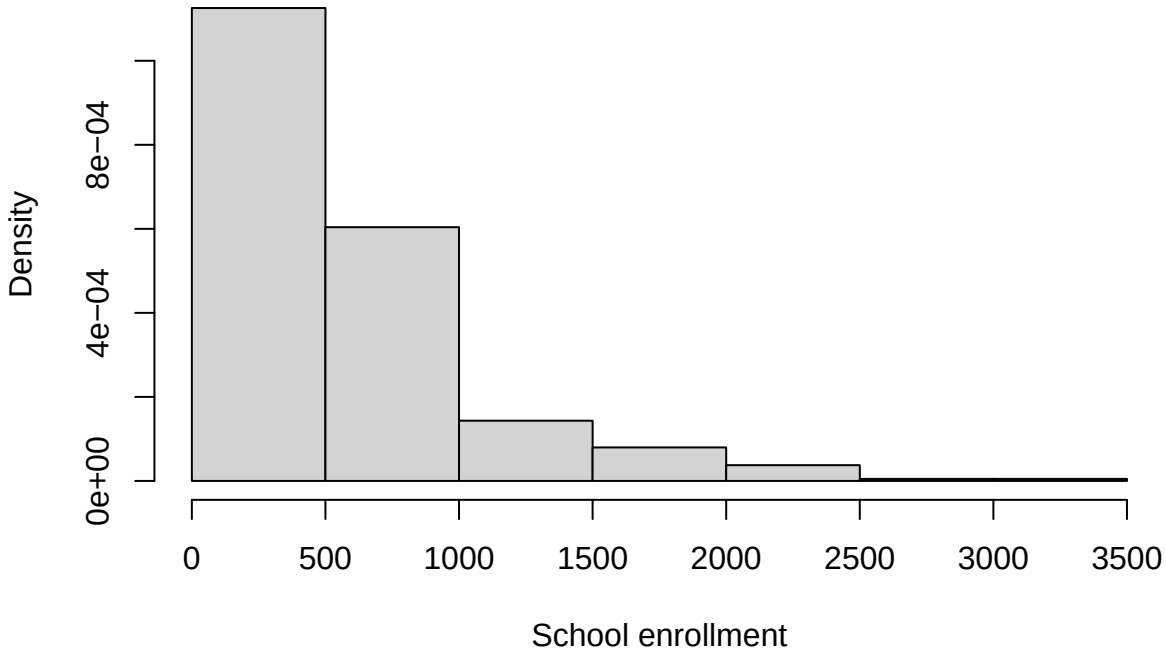
```
svytotal(~stype, srs_design)
##          total     SE
## stypeE 4397.74 196.00
## stypeH  774.25 142.85
## stypeM 1022.01 160.33
```

## Histogram of enrollment

`svyhist` accounts for the weights so that the proportion of the population in each bin is estimated correctly.

The `hist` command would not account for non-SRS of schools.

```
svyhist(~enroll, design = strat_design, xlab = "School enrollment", main = "")
```




## Cluster Sampling

### A one-stage cluster sample

- There are 757 districts in total, and 15 districts are sampled.
- The weight is calculated from $w_k = \pi_k^{-1}$.
- The sampling weights in `apiclus1` are incorrect but are as obtained from UCLA.
- So weight should be $757/15 = 50.47$ but reported as (`pw`) 33.847. The sampling probabilities are $15/757 = 0.01982$.
- All schools within the districts selected were then sampled, to give 183 schools in total.

Below we define a survey object with the `id` variable indicating that the PSUs are districts, as indicated by `dnum`. `fpc` gives the size of the population (number of districts) from which sampling was carried out, here 757.

```
clus1_design <- svydesign(id = ~dnum, data = apiclus1, fpc = ~fpc)
clus1_design
## 1 - level Cluster Sampling design
## With (15) clusters.
## svydesign(id = ~dnum, data = apiclus1, fpc = ~fpc)
dim(clus1_design)
## [1] 183  39
table(apiclus1$fpc)
##
## 757
## 183
```

```
summary(clus1_design)
## 1 - level Cluster Sampling design
## With (15) clusters.
## svydesign(id = ~dnum, data = apiclus1, fpc = ~fpc)
```

```
## Probabilities:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.01982 0.01982 0.01982 0.01982 0.01982 0.01982
## Population size (PSUs): 757
## Data variables:
##  [1] "cds"      "stype"    "name"     "sname"    "snum"     "dname"
##  [7] "dnum"     "cname"    "cnum"     "flag"     "pcttest"  "api00"
## [13] "api99"    "target"   "growth"   "sch.wide" "comp.imp" "both"
## [19] "awards"   "meals"    "ell"      "yr.rnd"   "mobility" "acs.k3"
## [25] "acs.46"   "acs.core" "pct.resp" "not.hsg"  "hsg"      "some.col"
## [31] "col.grad" "grad.sch" "avg.ed"   "full"     "emer"     "enroll"
## [37] "api.stu"  "fpc"      "pw"
```

```
head(apiclus1, n = 3)
##             cds stype            name                sname snum
## 1 01612910137588     H San Leandro Hig     San Leandro High  236
## 2 01612916002372     E Garfield Elemen  Garfield Elementary  237
## 3 01612916002398     E Jefferson Eleme Jefferson Elementary  238
##                 dname dnum   cname cnum flag pcttest api00 api99 target growth
## 1 San Leandro Unified  637 Alameda    1   NA       97   608   562     12     46
## 2 San Leandro Unified  637 Alameda    1   NA      100   684   554     12    130
## 3 San Leandro Unified  637 Alameda    1   NA      100   612   528     14     84
##   sch.wide comp.imp both awards meals ell yr.rnd mobility acs.k3 acs.46
## 1      Yes      Yes  Yes    Yes    19  22     No       15     NA     NA
## 2      Yes      Yes  Yes    Yes    39  23     No       23     19     30
## 3      Yes      Yes  Yes    Yes    39  27     No       25     21     30
##   acs.core pct.resp not.hsg hsg some.col col.grad grad.sch avg.ed full emer
## 1       27       90      14  22       27       30        6   2.93   82   23
## 2       NA       85       8  22       38       24        8   3.02   79   21
## 3       NA       95      12  24       40       18        6   2.83   72   31
##   enroll api.stu fpc     pw
## 1   1689    1358 757 33.847
## 2    288     223 757 33.847
## 3    294     220 757 33.847
```

There are 757 districts (PSUs) and 15 were sampled.

```
length(unique(apipop$dname))
## [1] 757
length(unique(apiclus1$dname))
## [1] 15
dim(apiclus1)
## [1] 183  39
table(apiclus1$pw)
##
## 33.846996307373
##             183
table(apiclus1$dnum)
##
##  61 135 178 197 255 406 413 437 448 510 568 637 716 778 815
##  13  34   4  13  16   2   1   4  12  21   9  11  37   2   4
```

These are 183 schools in the cluster sample (the sum of the last row).

### A two-stage cluster sample

In the two-stage cluster sample, 40 school districts are sampled and then up to five schools are sampled from each of the sampled districts.

We specify the design as below.

```
clus2_design <- svydesign(id = ~dnum +
    snum, fpc = ~fpc1 +
    fpc2, data = apiclus2)
clus2_design
## 2 - level Cluster Sampling design
## With (40, 126) clusters.
## svydesign(id = ~dnum + snum, fpc = ~fpc1 + fpc2, data = apiclus2)
```

### Exercises

- For both data from the 1-stage and 2-stage cluster designs, create histograms of the API00 variable using the `svyhist` and `hist` functions.
- For both data from the 1-stage and 2-stage cluster desgins, fit linear models regressing API00 on API99 using the `svyglm` and `lm` functions.
- What do you expect to see, i.e., when are there differences and when are there no differences?

## BRFSS Data

### Read in Data

`BRFSS` contains the full BRFSS dataset with 16,283 observations:

- `diab2` variable is the binary indicator of Type II diabetes
- `strata` is the strata indicator and
- `rwt_llcp` is the final design weight.

For the purpose of this analysis, we first remove records with missing HRA code or diabetes status from this dataset.

```
library(ggplot2)
library(patchwork)
library(SUMMER)
data(BRFSS)
data(KingCounty)
BRFSS <- subset(BRFSS, !is.na(BRFSS$diab2))
BRFSS <- subset(BRFSS, !is.na(BRFSS$hracode))
```

### Design object and direct estimates

We have stratified, disproportionate sampling, so note the arguments:

- `weights`
- `strata`

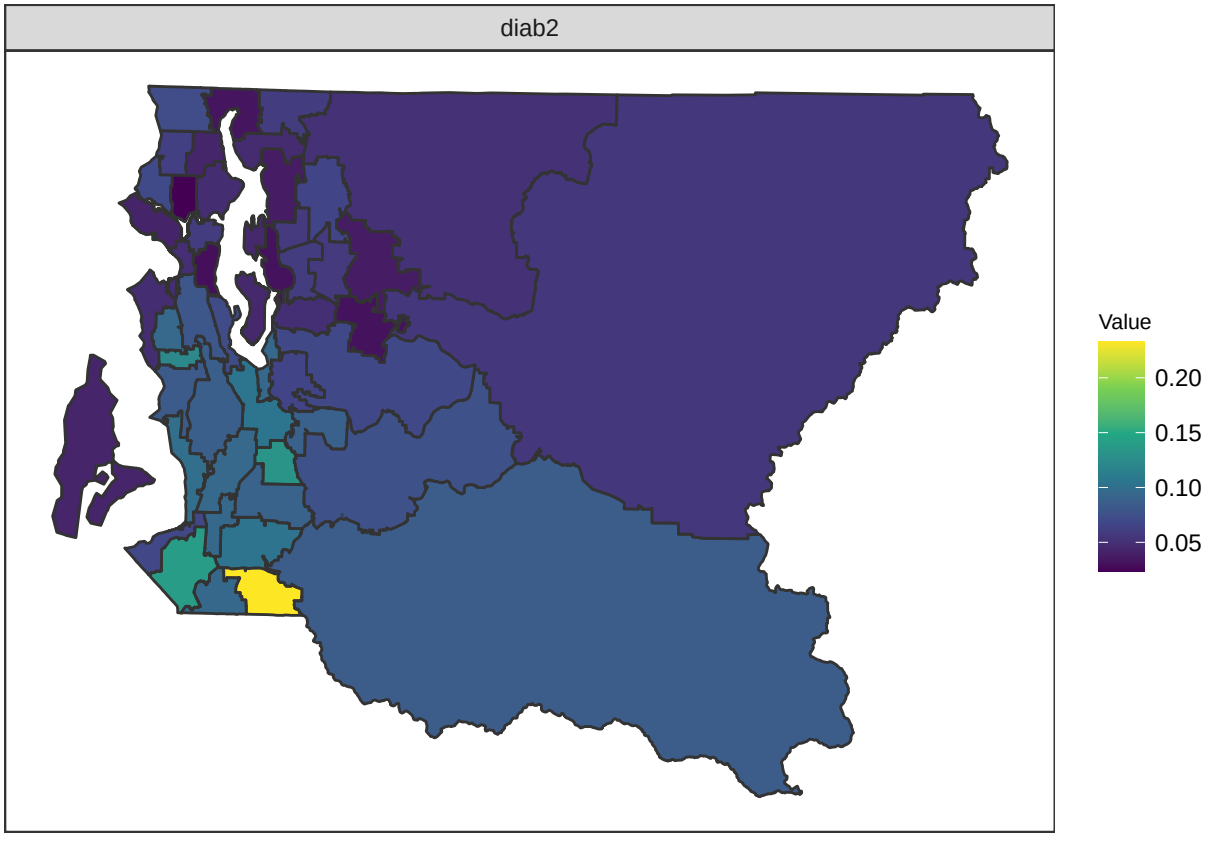
We then calculate the direct (weighted) esimates using the `survey` package.

```
library(survey)
design <- svydesign(ids = ~1, weights = ~rwt_llcp, strata = ~strata, data = BRFSS)
direct <- svyby(~diab2, ~hracode, design, svymean)
head(direct, n = 5)
##                                                hracode      diab2         se
## Auburn-North                                Auburn-North 0.10403154 0.02147752
## Auburn-South                                Auburn-South 0.23293289 0.04897800
## Ballard                                         Ballard 0.07047572 0.02225241
## Beacon/Gtown/S.Park                  Beacon/Gtown/S.Park 0.08083033 0.02603522
## Bear Creek/Carnation/Duvall Bear Creek/Carnation/Duvall 0.05166773 0.01190146
toplotB <- data.frame(direct)
```

### Maps

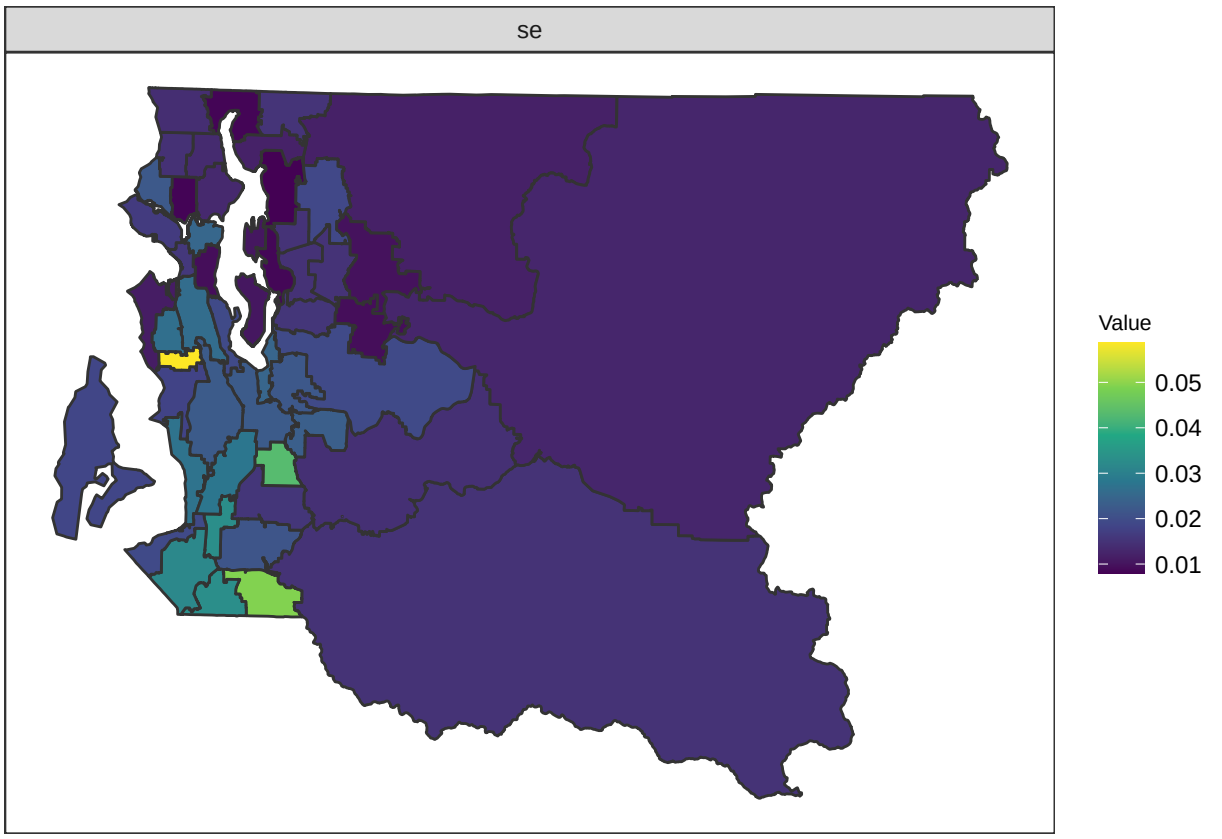
Direct estimates:

```
mapPlot(data = toplotB, geo = KingCounty, variables = c("diab2"), by.data = "hracode",
    by.geo = "HRA2010v2_")
```



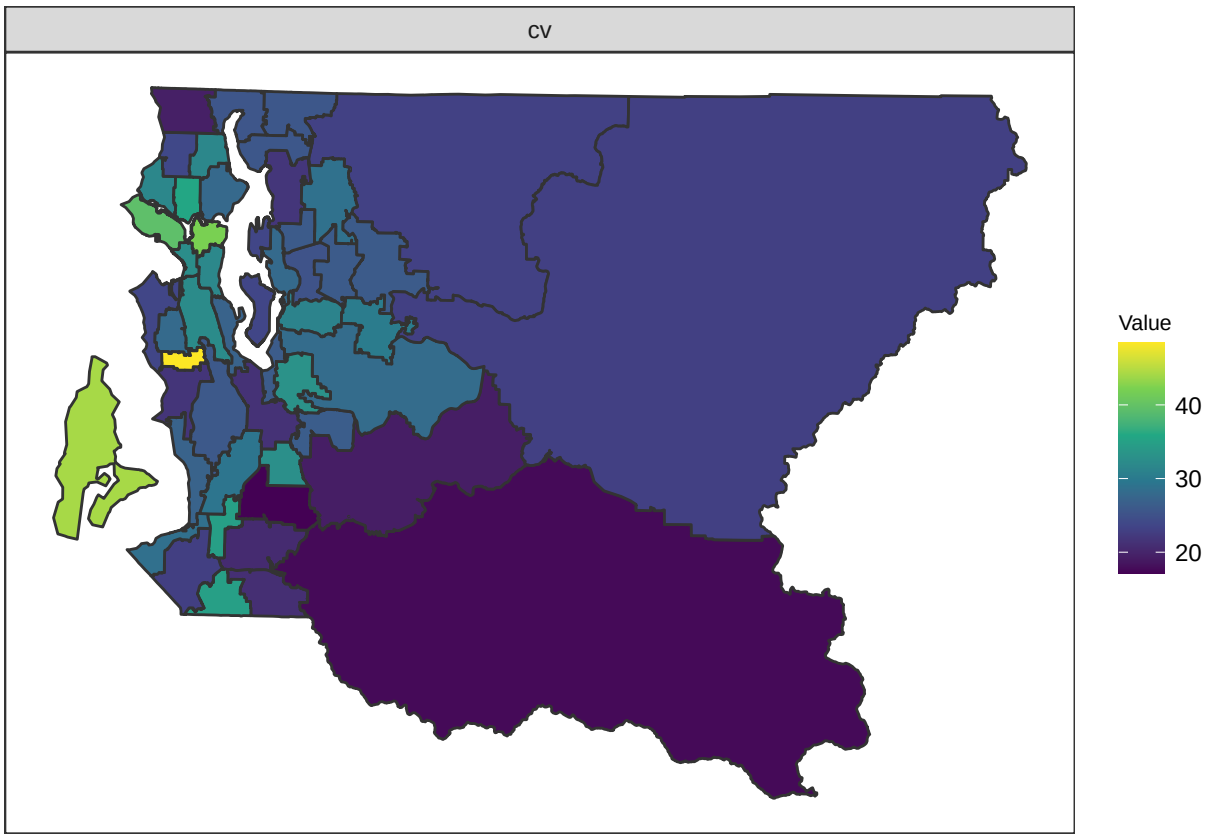

Standard errors:

```
mapPlot(data = toplotB, geo = KingCounty, variables = c("se"), by.data = "hracode",
    by.geo = "HRA2010v2_")
```




Coefficients of variation (100 × SD/Mean):

```
toplotB$cv <- 100 * toplotB$se/toplotB$diab2
mapPlot(data = toplotB, geo = KingCounty, variables = c("cv"), by.data = "hracode",
    by.geo = "HRA2010v2_")
```

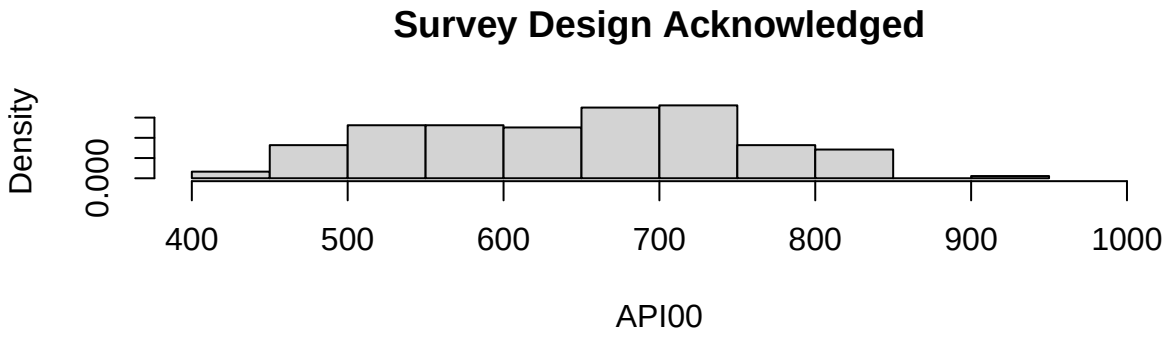


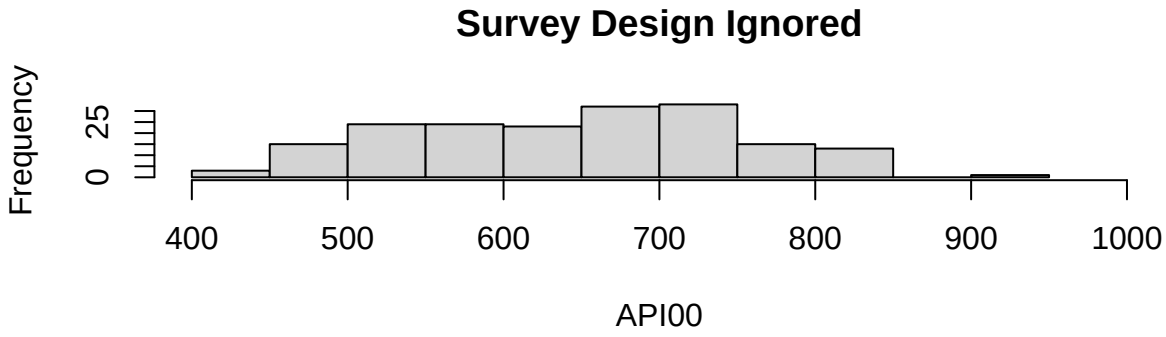

The CV is often used in survey sampling, with values over (say) 30% being deemed unacceptably large.

## Solutions to Exercises

1-stage cluster sampling: Histograms of API00 ignoring and acknowledging the design.

```
par(mfrow = c(2, 1))
svyhist(~api00, design = clus1_design, xlim = c(400, 1000), main = "Survey Design Acknowledged",
    xlab = "API00")
hist(apiclus1$api00, xlim = c(400, 1000), main = "Survey Design Ignored", xlab = "API00")
```



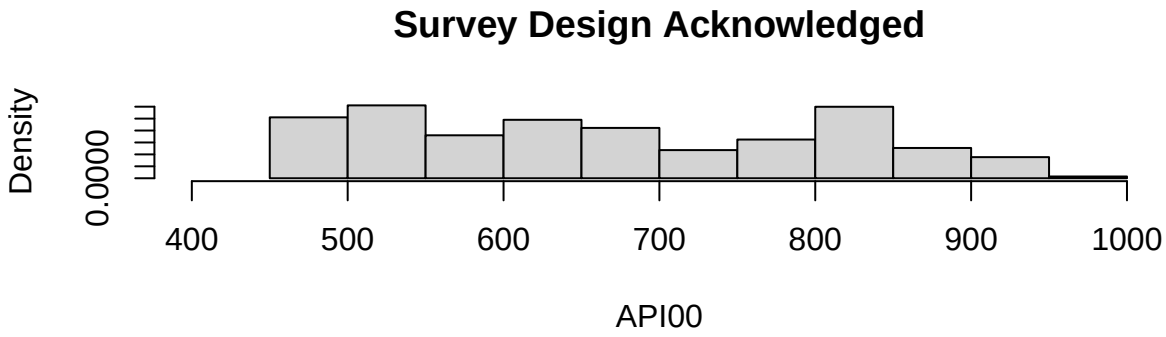



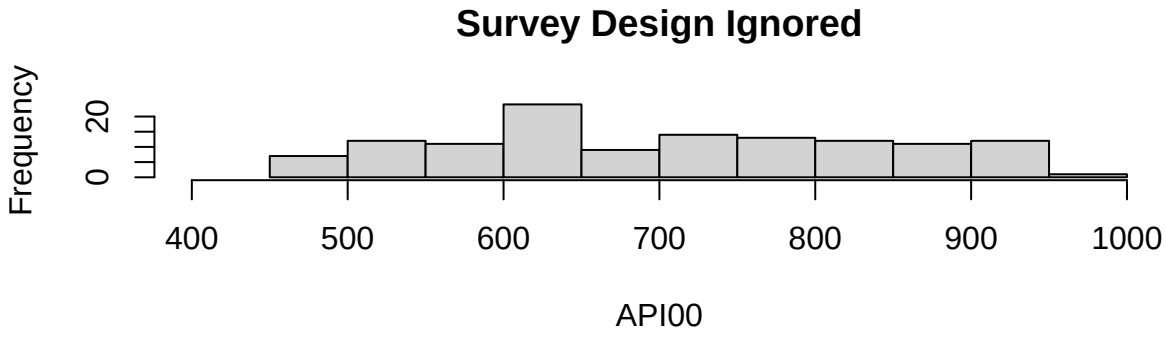

- Makes no difference since the weights are equal here.

```
table(apiclus1$pw)
##
## 33.846996307373
##             183
```

2-stage cluster sampling: Histograms of API00 ignoring and acknowledging the design.

```
par(mfrow = c(2, 1))
svyhist(~api00, design = clus2_design, xlim = c(400, 1000), main = "Survey Design Acknowledged",
    xlab = "API00")
hist(apiclus2$api00, xlim = c(400, 1000), main = "Survey Design Ignored", xlab = "API00")
```




- Makes a difference since the weights are not equal here:

```
table(apiclus2$pw)
##
## 18.925  22.71 26.495 34.065 41.635  52.99 105.98 136.26 272.52
##     81      5     10      5      5      5      5      5      5
```

Now regression:

```
lmc1 <- svyglm(api00 ~ api99, design = clus1_design)
summary(lmc1)
##
## Call:
## svyglm(formula = api00 ~ api99, design = clus1_design)
##
## Survey design:
## svydesign(id = ~dnum, data = apiclus1, fpc = ~fpc)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 95.28483   14.98802   6.357 2.51e-05 ***
## api99        0.90429    0.02361  38.301 9.38e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 768.7169)
##
## Number of Fisher Scoring iterations: 2
lmc1un <- lm(apiclus1$api00 ~ apiclus1$api99)
summary(lmc1un)
##
```

```
## Call:
## lm(formula = apiclus1$api00 ~ apiclus1$api99)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -68.252 -19.671   0.468  14.674  87.738
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    95.28483   11.27337   8.452 9.23e-15 ***
## apiclus1$api99  0.90429    0.01826  49.518  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.8 on 181 degrees of freedom
## Multiple R-squared:  0.9313, Adjusted R-squared:  0.9309
## F-statistic:  2452 on 1 and 181 DF,  p-value: < 2.2e-16
```

Notes:

- the estimates are the same (the weights are equal here).
- the standard errors are different because the cluster design gives a different amount of information.

```
lmc2 <- svyglm(api00 ~ api99, design = clus2_design)
summary(lmc2)
##
## Call:
## svyglm(formula = api00 ~ api99, design = clus2_design)
##
## Survey design:
## svydesign(id = ~dnum + snum, fpc = ~fpc1 + fpc2, data = apiclus2)
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.78909   11.90600   3.594 0.000923 ***
## api99        0.97363    0.01578  61.710  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 647.3376)
##
## Number of Fisher Scoring iterations: 2
lmc2un <- lm(apiclus2$api00 ~ apiclus2$api99)
summary(lmc2un)
##
## Call:
## lm(formula = apiclus2$api00 ~ apiclus2$api99)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -60.235 -17.099  -1.295  13.927  70.115
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)     54.2435    12.2309   4.435 2.01e-05 ***
## apiclus2$api99   0.9588     0.0177  54.176  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.12 on 124 degrees of freedom
## Multiple R-squared:  0.9595, Adjusted R-squared:  0.9591
## F-statistic:  2935 on 1 and 124 DF,  p-value: < 2.2e-16
```

Notes:

- The estimates are different (because the weights differ for this design).
- The standard errors are different because the cluster design gives a different amount of information. Often cluster samples have higher standard errors, but not always as it depends on sampling variability (and the within-cluster dependency, i.e. the dependency of outcomes on units within the same cluster).