A survey analysis example

Thomas Lumley

December 9, 2005

This document provides a simple example analysis of a survey data set, a subsample from the California Academic Performance Index, an annual set of tests used to evaluate California schools. The API website, including the original data files are at `http://api.cde.ca.gov`. The subsample was generated as a teaching example by Academic Technology Services at UCLA and was obtained from `http://www.ats.ucla.edu/stat/stata/Library/svy_survey.htm`.

We have a cluster sample in which 15 school districts were sampled and then all schools in each district. This is in the data frame `apiclus1`, loaded with `data(api)`. The two-stage sample is defined by the sampling unit (`dnum`) and the population size(`fpc`). Sampling weights are computed from the population sizes, but could be provided separately.

```
> data(api)
> dclus1 <- svydesign(id = ~dnum, weights = ~pw, data = apiclus1,
+     fpc = ~fpc)
```

The `svydesign` function returns an object containing the survey data and metadata.

```
> summary(dclus1)

1 - level Cluster Sampling design
With (15) clusters.
svydesign(id = ~dnum, weights = ~pw, data = apiclus1, fpc = ~fpc)
Probabilities:
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.02954 0.02954 0.02954 0.02954 0.02954 0.02954
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"
```

We can compute summary statistics to estimate the mean, median, and quartiles of the Academic Performance Index in the year 2000, the number of elementary, middle, and high schools in the state, the total number of students, and the proportion who took the test. Each function takes a formula object describing the variables and a survey design object containing the data.

```
> svymean(~api00, dclus1)

        mean     SE
api00 644.17 23.542

> svyquantile(~api00, dclus1, quantile = c(0.25, 0.5, 0.75), ci = TRUE)

$quantiles
        0.25 0.5  0.75
api00 551.75 652 717.5

$CIs
, , api00

           0.25      0.5     0.75
(lower 493.2835 564.3250 696.0000
upper) 622.6495 710.8375 761.1355

> svytotal(~stype, dclus1)

         total      SE
stypeE 4873.97 1333.32
stypeH  473.86  158.70
stypeM  846.17  167.55

> svytotal(~enroll, dclus1)

         total     SE
enroll 3404940 932235

> svyratio(~api.stu, ~enroll, dclus1)

Ratio estimator: svyratio.survey.design2(~api.stu, ~enroll, dclus1)
Ratios=
           enroll
api.stu 0.8497087
SEs=
             enroll
api.stu 0.008386297
```

The ordinary R subsetting functions `[` and `subset` work correctly on these survey objects, carrying along the metadata needed for valid standard errors. Here we compute the proportion of high school students who took the test

```
> svyratio(~api.stu, ~enroll, design = subset(dclus1, stype ==
+     "H"))

Ratio estimator: svyratio.survey.design2(~api.stu, ~enroll, design = subset(dclus1,
    stype == "H"))
Ratios=
           enroll
api.stu 0.8300683
SEs=
            enroll
api.stu 0.01472607
```

The warnings referred to in the output occured because several school districts have only one high school sampled, making the second stage standard error estimation unreliable.

Summary statistics for subsets can also be computed with `svyby`. Here we compute the average proportion of “English language learners” and of students eligible for subsidized school meals for elementary, middle, and high schools

```
> svyby(~ell + meals, ~stype, design = dclus1, svymean)

  stype statistic.ell statistic.meals      SE1      SE2
E     E      29.69444        53.09028 1.411617 7.070399
H     H      15.00000        37.57143 5.347065 5.912262
M     M      22.68000        43.08000 2.952862 6.017110
```

Regression models show that these socieconomic variables predict API score and whether the school achieved its API target

```
> regmodel <- svyglm(api00 ~ ell + meals, design = dclus1)
> logitmodel <- svyglm(I(sch.wide == "Yes") ~ ell + meals, design = dclus1,
+     family = quasibinomial())
> summary(regmodel)

Call:
svyglm(api00 ~ ell + meals, design = dclus1)

Survey design:
svydesign(id = ~dnum, weights = ~pw, data = apiclus1, fpc = ~fpc)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 817.1823    18.6709  43.768   <2e-16 ***
ell          -0.5088     0.3259  -1.561    0.120
meals        -3.1456     0.3018 -10.423   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for gaussian family taken to be 3161.207)

Number of Fisher Scoring iterations: 2

> summary(logitmodel)

Call:
svyglm(I(sch.wide == "Yes") ~ ell + meals, design = dclus1, family = quasibinomial())

Survey design:
svydesign(id = ~dnum, weights = ~pw, data = apiclus1, fpc = ~fpc)

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.899557   0.509900   3.725 0.000261 ***
ell          0.039925   0.012442   3.209 0.001578 **
meals       -0.019115   0.008825  -2.166 0.031630 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasibinomial family taken to be 0.9627734)

Number of Fisher Scoring iterations: 5
```

We can calibrate the sampling using the statewide total for the previous year's API

```
> gclus1 <- calibrate(dclus1, formula = ~api99, population = c(6194,
+     3914069))
```

which improves estimation of some quantities

```
> svymean(~api00, gclus1)

        mean     SE
api00 666.72 3.2959

> svyquantile(~api00, gclus1, quantile = c(0.25, 0.5, 0.75), ci = TRUE)

$quantiles
          0.25     0.5     0.75
api00 592.0652 681.181 736.5414

$CIs
, , api00

           0.25      0.5     0.75
(lower 553.0472 663.3175 721.1432
upper) 617.0915 696.8528 755.2959
```

```
> svytotal(~stype, gclus1)

         total     SE
stypeE 4881.77 302.15
stypeH  463.35 183.03
stypeM  848.88 194.76

> svytotal(~enroll, gclus1)

         total     SE
enroll 3357372 243227

> svyratio(~api.stu, ~enroll, gclus1)

Ratio estimator: svyratio.survey.design2(~api.stu, ~enroll, gclus1)
Ratios=
           enroll
api.stu 0.8506941
SEs=
             enroll
api.stu 0.008674888
```