Estimates in subpopulations.

Thomas Lumley

May 15, 2006

Estimating a mean or total in a subpopulation (domain) from a survey, eg the mean blood pressure in women, is not done simply by taking the subset of data in that subpopulation and pretending it is a new survey. This approach would give correct point estimates but incorrect standard errors.

The standard way to derive domain means is as ratio estimators. I think it is easier to derive them as regression coefficients. These derivations are not important for R users, since subset operations on survey design objects automatically do the necessary adjustments, but they may be of interest. The various ways of constructing domain mean estimators are useful in quality control for the survey package, and some of the examples here are taken from `survey/tests/domain.R`.

Suppose that in the artificial `fpc` data set we want to estimate the mean of `x` when `x>4`.

```
> library(survey)
> data(fpc)
> dfpc <- svydesign(id = ~psuid, strat = ~stratid, weight = ~weight,
+     data = fpc, nest = TRUE)
> dsub <- subset(dfpc, x > 4)
> svymean(~x, design = dsub)
```

```
   mean     SE
x 6.195 0.7555
```

The `subset` function constructs a survey design object with information about this subpopulation and `svymean` computes the mean. The same operation can be done for a set of subpopulations with `svyby`.

```
> svyby(~x, ~I(x > 4), design = dfpc, svymean)
```

```
      I(x > 4) statistics.x      se.x
FALSE    FALSE     3.314286 0.3117042
TRUE      TRUE     6.195000 0.7555129
```

In a regression model with a binary covariate Z and no intercept, there are two coefficients that estimate the mean of the outcome variable in the subpopulations with $Z = 0$ and $Z = 1$, so we can construct the domain mean estimator by regression.

```
> summary(svyglm(x ~ I(x > 4) + 0, design = dfpc))

Call:
svyglm(x ~ I(x > 4) + 0, design = dfpc)

Survey design:
svydesign(id = ~psuid, strat = ~stratid, weight = ~weight, data = fpc,
    nest = TRUE)

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
I(x > 4)FALSE   3.3143     0.3117   10.63 0.000127 ***
I(x > 4)TRUE    6.1950     0.7555    8.20 0.000439 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

(Dispersion parameter for gaussian family taken to be 2.557379)

Number of Fisher Scoring iterations: 2
```

Finally, the classical derivation of the domain mean estimator is as a ratio where the numerator is X for observations in the domain and 0 otherwise and the denominator is 1 for observations in the domain and 0 otherwise

```
> svyratio(~I(x * (x > 4)), ~as.numeric(x > 4), dfpc)

Ratio estimator: svyratio.survey.design2(~I(x * (x > 4)), ~as.numeric(x > 4),
    dfpc)
Ratios=
               as.numeric(x > 4)
I(x * (x > 4))             6.195
SEs=
               as.numeric(x > 4)
I(x * (x > 4))         0.7555129
```

The estimator is implemented by setting the sampling weight to zero for observations not in the domain. For most survey design objects this allows a reduction in memory use, since only the number of zero weights in each sampling unit needs to be kept. For more complicated survey designs, such as post-stratified designs, all the data are kept and there is no reduction in memory use.

More complex examples

Verifying that `svymean` agrees with the ratio and regression derivations is particularly useful for more complicated designs where published examples are less readily available.

This example shows calibration (GREG) estimators of domain means for the California Academic Performance Index (API).

```
> data(api)
> dclus1 <- svydesign(id = ~dnum, weights = ~pw, data = apiclus1,
+     fpc = ~fpc)
> pop.totals <- c("(Intercept)" = 6194, stypeH = 755, stypeM = 1018)
> gclus1 <- calibrate(dclus1, ~stype + api99, c(pop.totals, api99 = 3914069))
> svymean(~api00, subset(gclus1, comp.imp == "Yes"))

        mean     SE
api00 672.05 6.5182

> svyratio(~I(api00 * (comp.imp == "Yes")), ~as.numeric(comp.imp ==
+     "Yes"), gclus1)

Ratio estimator: svyratio.survey.design2(~I(api00 * (comp.imp == "Yes")), ~as.numeric(comp.i
    "Yes"), gclus1)
Ratios=
                              as.numeric(comp.imp == "Yes")
I(api00 * (comp.imp == "Yes"))                       672.049
SEs=
                              as.numeric(comp.imp == "Yes")
I(api00 * (comp.imp == "Yes"))                      6.518153

> summary(svyglm(api00 ~ comp.imp - 1, gclus1))

Call:
svyglm(api00 ~ comp.imp - 1, gclus1)

Survey design:
calibrate(dclus1, ~stype + api99, c(pop.totals, api99 = 3914069))

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
comp.impNo   649.706     12.563   51.72   <2e-16 ***
comp.impYes  672.049      6.518  103.10   <2e-16 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

(Dispersion parameter for gaussian family taken to be 10519.86)

Number of Fisher Scoring iterations: 2
```

Two-stage samples with full finite-population corrections

```
> data(mu284)
> dmu284 <- svydesign(id = ~id1 + id2, fpc = ~n1 + n2, data = mu284)
> svymean(~y1, subset(dmu284, y1 > 40))
```

```
    mean     SE
y1 48.69 1.1088
```

```
> svyratio(~I(y1 * (y1 > 40)), ~as.numeric(y1 > 40), dmu284)
```

```
Ratio estimator: svyratio.survey.design2(~I(y1 * (y1 > 40)), ~as.numeric(y1 >
    40), dmu284)
Ratios=
                  as.numeric(y1 > 40)
I(y1 * (y1 > 40))            48.69014
SEs=
                  as.numeric(y1 > 40)
I(y1 * (y1 > 40))            1.108825
```

```
> summary(svyglm(y1 ~ I(y1 > 40) + 0, dmu284))
```

```
Call:
svyglm(y1 ~ I(y1 > 40) + 0, dmu284)

Survey design:
svydesign(id = ~id1 + id2, fpc = ~n1 + n2, data = mu284)

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
I(y1 > 40)FALSE   34.419      1.842   18.69 0.000334 ***
I(y1 > 40)TRUE    48.690      1.109   43.91  2.6e-05 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

(Dispersion parameter for gaussian family taken to be 27.96987)

Number of Fisher Scoring iterations: 2
```

Stratified two-phase sampling of children with Wilm's Tumor, estimating relapse probability for those older than 3 years (36 months) at diagnosis

```
> library("survival")
```

```
Loading required package: splines
```

```
> data(nwtco)
> nwtco$incc2 <- as.logical(with(nwtco, ifelse(rel | instit ==
+     2, 1, rbinom(nrow(nwtco), 1, 0.1))))
> dccs8 <- twophase(id = list(~seqno, ~seqno), strata = list(NULL,
+     ~interaction(rel, stage, instit)), data = nwtco, subset = ~incc2)
> svymean(~rel, subset(dccs8, age > 36))
```

```
       mean   SE
rel 0.16287 0.01
```

```
> svyratio(~I(rel * as.numeric(age > 36)), ~as.numeric(age > 36),
+     dccs8)

Ratio estimator: svyratio.twophase(~I(rel * as.numeric(age > 36)), ~as.numeric(age >
    36), dccs8)
Ratios=
                               as.numeric(age > 36)
I(rel * as.numeric(age > 36))            0.1628708
SEs=
                               as.numeric(age > 36)
I(rel * as.numeric(age > 36))          0.009994368

> summary(svyglm(rel ~ I(age > 36) + 0, dccs8))

Call:
svyglm(rel ~ I(age > 36) + 0, dccs8)

Survey design:
twophase(id = list(~seqno, ~seqno), strata = list(NULL, ~interaction(rel,
    stage, instit)), data = nwtco, subset = ~incc2)

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
I(age > 36)FALSE 0.116861   0.009087   12.86   <2e-16 ***
I(age > 36)TRUE  0.162871   0.009994   16.30   <2e-16 ***
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

(Dispersion parameter for gaussian family taken to be 0.1212451)

Number of Fisher Scoring iterations: 2
```