Obsolete formulas for two-phase variances

Thomas Lumley

April 27, 2019

This document explains the computation of variances for totals in two-phase designs before version 3.15, or using `method=approx`. Since version 3.15 the variances are computed directly using a sparse-matrix representation of the covariance of sampling indicators, and agree exactly with the formulas in Section 9.3 of Särndal, Swensson, and Wretman. Variances for other statistics are computed by the delta-method from the variance of the total of the estimating functions.

The variance formulas come from conditioning on the sample selected in the first phase

$$\text{var}[\hat{T}] = E\left[\text{var}\left[\hat{T}|\text{phase 1}\right]\right] + \text{var}\left[E\left[\hat{T}|\text{phase 1}\right]\right]$$

The first term is estimated by the variance of $\hat{T}$ considering the phase one sample as the fixed population, and so uses the same computations as any single-phase design. The second term is the variance of $\hat{T}$ if complete data were available for the phase-one sample. This takes a little more work.

The variance computations for a stratified, clustered, multistage design involve recursively computing a within-stratum variance for the total over sampling units at the next stage. That is, we want to compute

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})$$

where X_i are π-expanded observations, perhaps summed over sampling units. A natural estimator of s^2 when only some observations are present in the phase-two sample is

$$\hat{s}^2 = \frac{1}{n-1}\sum_{i=1}^{n}\frac{R_i}{\pi_i}(X_i - \hat{\bar{X}})$$

where π_i is the probability that X_i is available and R_i is the indicator that X_i is available. We also need an estimator for $\bar{X}$, and a natural one is

$$\hat{\bar{X}} = \frac{1}{n}\sum_{i=1}^{n}\frac{R_i}{\pi_i}X_i$$

This is not an unbiased estimator of s^2 unless $\hat{\bar{X}} = \bar{X}$, but the bias is of order $O(n_2^{-1})$ where $n_2 = \sum_i R_i$ is the number of phase-two observations.

If the phase-one design involves only a single stage of sampling then X_i is Y_i/p_i, where Y_i is the observed value and p_i is the phase-one sampling probability. For multistage phase-one designs (not yet implemented) X_i will be more complicated, but still feasible to automate.

This example shows the unbiased phase-one estimate (from Takahiro Tsuchiya) and the estimate I use, in a situation where the phase two sample is quite small.

First we read the data

```
rei<-read.table(textConnection(
"  id   N n.a h n.ah n.h   sub  y
1   1 300  20 1   12   5  TRUE  1
2   2 300  20 1   12   5  TRUE  2
3   3 300  20 1   12   5  TRUE  3
4   4 300  20 1   12   5  TRUE  4
5   5 300  20 1   12   5  TRUE  5
6   6 300  20 1   12   5 FALSE NA
7   7 300  20 1   12   5 FALSE NA
8   8 300  20 1   12   5 FALSE NA
9   9 300  20 1   12   5 FALSE NA
10 10 300  20 1   12   5 FALSE NA
11 11 300  20 1   12   5 FALSE NA
12 12 300  20 1   12   5 FALSE NA
13 13 300  20 2    8   3  TRUE  6
14 14 300  20 2    8   3  TRUE  7
15 15 300  20 2    8   3  TRUE  8
16 16 300  20 2    8   3 FALSE NA
17 17 300  20 2    8   3 FALSE NA
18 18 300  20 2    8   3 FALSE NA
19 19 300  20 2    8   3 FALSE NA
20 20 300  20 2    8   3 FALSE NA
"), header=TRUE)
```

Now, construct a two-phase design object and compute the total of `y`

```
> library(survey)
> des.rei <- twophase(id=list(~id,~id), strata=list(NULL,~h),
+                     fpc=list(~N,NULL), subset=~sub, data=rei)
> tot<- svytotal(~y, des.rei)
```

The unbiased estimator is given by equation 9.4.14 of Särndal, Swensson, & Wretman.

```
> rei$w.ah <- rei$n.ah / rei$n.a
> a.rei <- aggregate(rei, by=list(rei$h), mean, na.rm=TRUE)
> a.rei$S.ysh <- tapply(rei$y, rei$h, var, na.rm=TRUE)
> a.rei$y.u <- sum(a.rei$w.ah * a.rei$y)
> a.rei$f<-with(a.rei, n.a/N)
> a.rei$delta.h<-with(a.rei, (1/n.h)*(n.a-n.ah)/(n.a-1))
> Vphase1<-with(a.rei, sum(N*N*((1-f)/n.a)*( w.ah*(1-delta.h)*S.ysh+ ((n.a)/(n.a-1))*w.ah*(y-y.u)^2
```

The phase-two contributions (not shown) are identical. The phase-one contributions are quite close

```
> Vphase1

[1] 24072.63

> attr(vcov(tot),"phases")$phase1

         [,1]
[1,] 24072.63
```