Survey examples from UCLA ATS

Thomas Lumley

February 22, 2005

Academic Technology Services at UCLA has a number of excellent web pages showing how to perform various analyses in popular statistical packages. Here I show how some of their survey analyses, linked from `http://www.ats.ucla.edu/stat/survey/survey_howtochoose.htm`, can be performed in R. This also provides some verification that R is producing the right answers. The data are taken from *Sampling of Populations* by Levy and Lemeshow, and from *Practical Methods for Design and Analysis of Complex Surveys* by Lehtonen and Pakhinen. The ATS web site provides SAS files for the Levy & Lemeshow data sets; the analyses in this file show R reading from Stata versions converted using Stat/Transfer.

```
> library(survey)
> library(foreign)
```

1 Stratified sampling

The `hospital` data set included in the survey package is analysed by ATS as an example of stratified sampling. SUDAAN results are at `http://www.ats.ucla.edu/stat/sudaan/faq/svy_sudaan_strrs.htm` and WesVar results at `http://www.ats.ucla.edu/stat/wesvar/faq/strrs/svy_wv_strrs.htm`

```
> data(hospital)
> dstr <- svydesign(id = ~1, strata = ~oblevel, fpc = ~tothosp,
+     weight = ~weighta, data = hospital)
> svymean(~births, dstr)

         mean     SE
births 1164.4 215.28

> svytotal(~births, dstr)

        total    SE
births 183983 34014

> svyby(~births, ~oblevel, dstr, svymean, keep.var = TRUE)
```

```
  oblevel statistic.births        SE
1       1            355.5  63.56802
2       2           1183.0 334.01701
3       3           3055.0 441.67075

> svyby(~births, ~oblevel, dstr, svytotal, keep.var = TRUE)

  oblevel statistic.births        SE
1       1            14931  2669.857
2       2           117117 33067.684
3       3            51935  7508.403

> rstr <- as.svrepdesign(dstr)
> svymean(~births, rstr)

         mean     SE
births 1164.4 215.28

> svytotal(~births, rstr)

        total    SE
births 183983 34014

> svyby(~births, ~oblevel, rstr, svymean, keep.var = TRUE)

  oblevel statistic.births        SE
1       1            355.5  63.56802
2       2           1183.0 334.01701
3       3           3055.0 441.67075

> svyby(~births, ~oblevel, rstr, svytotal, keep.var = TRUE)

  oblevel statistic.births        SE
1       1            14931  2669.857
2       2           117117 33067.684
3       3            51935  7508.403
```

This example is from p74 of Lehtonen & Pakhinen, with SUDAAN analysis at `http://www.ats.ucla.edu/stat/sudaan/faq/svy_sudaan_strrs.htm`

```
> unemp <- read.table(textConnection(" id str clu wt ue91 lab91 fpc
  1 1 1 1.75 4123 33786 7
  2 1 2 1.75 666 6016 7
  3 1 4 1.75 760 5919 7
  4 1 6 1.75 457 3022 7
  5 2 21 6.25 61 573 25
  6 2 25 6.25 262 1737 25
  7 2 26 6.25 331 2543 25
  8 2 27 6.25 98 545 25
"),
+     header = TRUE)
> dunemp <- svydesign(id = ~clu, strata = ~str, weight = ~wt, fpc = ~fpc,
+     data = unemp)
> svymean(~ue91, dunemp)

       mean     SE
ue91 475.33 133.73

> svyratio(~ue91, ~lab91, dunemp)

Ratio estimator: svyratio.survey.design2(~ue91, ~lab91, dunemp)
Ratios=
         lab91
ue91 0.1277788
SEs=
           lab91
ue91 0.003173564
```

2 Certainty PSUs

These data are from p60 of Lehtonen & Pakhinen. We give three analyses. The first is the analysis shown for SUDAAN and SAS at the UCLA ATS web site. In this analysis the finite population correction for the data is not used and the certainty PSU is split into two pseudo-PSUs to fool the software

```
> unemp <- read.table(textConnection(" id str clu wt hou85 ue91 lab91
  1 2 1 0.5 26881 4123 33786
  2 2 1 0.5 26881 4123 33786
  3 1 10 1.004 9230 1623 13727
  4 1 4 1.893 4896 760 5919
  5 1 7 2.173 4264 767 5823
  6 1 32 2.971 3119 568 4011
  7 1 26 4.762 1946 331 2543
  8 1 18 6.335 1463 187 1448
  9 1 13 13.730 675 129 927
"),
+     header = TRUE)
> dunemp <- svydesign(id = ~id, strata = ~str, weight = ~wt, data = unemp)
> svymean(~ue91, dunemp)

       mean     SE
ue91 445.18 150.47

> svytotal(~ue91, dunemp)
```

```
     total     SE
ue91 15077 521.12
```

R can handle certainty PSUs by setting an option.

```
> unemp <- read.table(textConnection(" id str clu wt hou85 ue91 lab91
  1 2 1 1 26881 4123 33786
  3 1 10 1.004 9230 1623 13727
  4 1 4 1.893 4896 760 5919
  5 1 7 2.173 4264 767 5823
  6 1 32 2.971 3119 568 4011
  7 1 26 4.762 1946 331 2543
  8 1 18 6.335 1463 187 1448
  9 1 13 13.730 675 129 927
"),
+     header = TRUE)
> options(survey.lonely.psu = "certainty")
> dunemp <- svydesign(id = ~clu, strata = ~str, weight = ~wt, data = unemp)
> svymean(~ue91, dunemp)

       mean     SE
ue91 445.18 150.47

> svytotal(~ue91, dunemp)

     total     SE
ue91 15077 521.12

> options(survey.lonely.psu = "fail")
```

The most correct way to handle certainty PSUs is to supply finite population corrections that show the PSU has been sampled with certainty. In this example the population size is 32, so the population size for the stratum not including the PSU is 31. To match the UCLA ATS analyses, which did not use the population size, we have a second analysis that pretends the population is very large.

```
> unemp <- read.table(textConnection(" id str clu wt hou85 ue91 lab91
  1 2 1 1 26881 4123 33786
  3 1 10 1.004 9230 1623 13727
  4 1 4 1.893 4896 760 5919
  5 1 7 2.173 4264 767 5823
  6 1 32 2.971 3119 568 4011
  7 1 26 4.762 1946 331 2543
  8 1 18 6.335 1463 187 1448
  9 1 13 13.730 675 129 927
"),
+     header = TRUE)
> VERYLARGE <- 1e+10
> dunemp <- svydesign(id = ~clu, strata = ~str, weight = ~wt, data = unemp,
+     fpc = c(1, rep(31, 7)))
> svymean(~ue91, dunemp)

       mean     SE
ue91 445.18 132.39

> svytotal(~ue91, dunemp)

     total     SE
ue91 15077 458.53

> dunemp1 <- svydesign(id = ~clu, strata = ~str, weight = ~wt,
+     data = unemp, fpc = c(1, rep(VERYLARGE, 7)))
> svymean(~ue91, dunemp1)

       mean     SE
ue91 445.18 150.47

> svytotal(~ue91, dunemp1)

     total     SE
ue91 15077 521.12
```

3 One-stage cluster sampling

An example from p29 of Lehtonen and Pakhinen. The analysis in SUDAAN at `http://www.ats.ucla.edu/stat/sudaan/faq/svy_sudaan_oscluster.htm` agrees with R. The analysis in WesVar at `http://www.ats.ucla.edu/stat/wesvar/faq/oscluster/svy_wv_oscluster.htm` does not, particularly for the standard error of ratio estimator where the difference is about 9%. Most of this difference is because the analysis on the web site is using the wrong finite population correction. The remaining difference is because WesVar and R use slightly different variance formulas. If $\theta_{(i)}$ are the replicates and $\hat{\theta}$ is the point estimate, R uses a variance formula based on $\sum_i(\theta_{(i)} - \bar{\theta}_{(i)})^2$ where WesVar appears to use $\sum_i(\theta_{(i)} - \hat{\theta})^2$. The formulas are equivalent for linear statistics such as the mean and total. For nonlinear statistics such as the ratio it does not appear to be clear which is best.

```
> unemp <- read.table(textConnection("id str clu wt ue91 lab91
  1 1 2 4 666 6016
  2 1 2 4 528 3818
  3 1 2 4 760 5919
  4 1 2 4 187 1448
  5 1 8 4 129 927
  6 1 8 4 128 819
  7 1 8 4 331 2543
  8 1 8 4 568 4011
"),
+     header = TRUE)
> dunemp <- svydesign(id = ~clu, weight = ~wt, fpc = rep(32, 8),
+     data = unemp)
> svytotal(~ue91 + lab91, dunemp)

       total      SE
ue91   13188  3814.9
lab91 102004 34473.4

> svyratio(~ue91, ~lab91, dunemp)

Ratio estimator: svyratio.survey.design2(~ue91, ~lab91, dunemp)
Ratios=
         lab91
ue91 0.1292890
SEs=
           lab91
ue91 0.006295318

> runemp <- as.svrepdesign(dunemp)
> svytotal(~ue91 + lab91, runemp)

       total      SE
ue91   13188  3814.9
lab91 102004 34473.4

> svyratio(~ue91, ~lab91, runemp)

Ratio estimator: svyratio.svyrep.design(~ue91, ~lab91, runemp)
Ratios=
         lab91
ue91 0.1292890
SEs=
            [,1]
[1,] 0.007168699
```

In this example from Levy and Lemeshow, R again agrees with SUDAAN. R also agrees with WesVar on the means and totals; the web site unfortunately does not show results for the ratio.

```
> tab9 <- read.dta("tab9_1c.dta")
> dtab9 <- svydesign(id = ~devlpmnt, weight = ~wt1, fpc = ~m, data = tab9)
> svymean(~nge65 + nvstnrs + hhneedvn, dtab9)

          mean     SE
nge65    1.675 0.0194
nvstnrs  0.575 0.0194
hhneedvn 0.525 0.0194
```

```
> svytotal(~nge65 + nvstnrs + hhneedvn, dtab9)

         total     SE
nge65    167.5 1.9365
nvstnrs   57.5 1.9365
hhneedvn  52.5 1.9365

> svyratio(~nvstnrs, ~nge65, dtab9)

Ratio estimator: svyratio.survey.design2(~nvstnrs, ~nge65, dtab9)
Ratios=
            nge65
nvstnrs 0.3432836
SEs=
              nge65
nvstnrs 0.007592393

> rtab9 <- as.svrepdesign(dtab9)
> svymean(~nge65 + nvstnrs + hhneedvn, dtab9)

          mean     SE
nge65    1.675 0.0194
nvstnrs  0.575 0.0194
hhneedvn 0.525 0.0194

> svytotal(~nge65 + nvstnrs + hhneedvn, dtab9)

         total     SE
nge65    167.5 1.9365
nvstnrs   57.5 1.9365
hhneedvn  52.5 1.9365

> svyratio(~nvstnrs, ~nge65, dtab9)

Ratio estimator: svyratio.survey.design2(~nvstnrs, ~nge65, dtab9)
Ratios=
            nge65
nvstnrs 0.3432836
SEs=
              nge65
nvstnrs 0.007592393
```

4 Post-stratification

Of the packages covered by the ATS web site only SUDAAN and WesVar can perform post-stratified analyses. These examples are taken from `http://www.ats.ucla.edu/stat/sudaan/faq/svy_sudaan_post.htm`, which shows the results of two analyses using SUDAAN, and `http://www.ats.ucla.edu/stat/wesvar/faq/post/svy_wv_post.htm`, which uses WesVar.

First an example from Levy & Lemeshow, of a post-stratified sample from the records of a veterinary practice. R gives the same results as SUDAAN for this analysis.

```
> pets <- read.dta("dogscats.dta")
> dpets <- svydesign(id = ~1, weight = ~weight, data = pets, fpc = ~n)
> pspets <- postStratify(dpets, ~type, data.frame(type = c(1, 2),
+     Freq = c(850, 450)))
> svymean(~totexp, pspets)

         mean     SE
totexp 40.115 1.1635

> svytotal(~totexp, pspets)

       total     SE
totexp 52150 1512.5

> svyby(~totexp, ~type, design = pspets, svymean, keep.var = TRUE)

  type statistic.totexp       SE
1    1         49.85844 1.443690
2    2         21.71111 1.965050

> svyby(~totexp, ~type, design = pspets, svytotal)

  type statistic
1    1  42379.67
2    2   9770.00
```

The analysis in WesVar does not use the finite population correction. Standard errors in R differ from those in Wesvar in the fourth significant digit.

```
> pets <- read.dta("dogscats.dta")
> rpets <- as.svrepdesign(svydesign(id = ~1, weight = ~weight,
+     data = pets))
> pspets <- postStratify(rpets, ~type, data.frame(type = c(1, 2),
+     Freq = c(850, 450)))
> svymean(~totexp, pspets)

         mean    SE
totexp 40.115 1.212

> svytotal(~totexp, pspets)

       total     SE
totexp 52150 1575.6
```

```
> svyby(~totexp, ~type, design = pspets, svymean, keep.var = TRUE)

  type statistic.totexp       SE
1    1         49.85844 1.489378
2    2         21.71111 2.079409

> svyby(~totexp, ~type, design = pspets, svytotal)

  type statistic
1    1  42379.67
2    2   9770.00
```

This example from Lehtonen & Pakhinen examines unemployment in a one-stage cluster sample. R and SUDAAN give the same results.

```
> unemp <- read.table(textConnection("id str clu wt ue91 lab91 poststr gwt postwt sruv srcvs
  1 1 1 4 4123 33786 1 .5833 2.333 .25 .43
  2 1 4 4 760 5919 1 .5833 2.333 .25 .43
  3 1 5 4 721 4930 1 .5833 2.333 .25 .43
  4 1 15 4 142 675 2 1.2500 5.0000 .25 .20
  5 1 18 4 187 1448 2 1.2500 5.0000 .25 .20
  6 1 26 4 331 2543 2 1.2500 5.0000 .25 .20
  7 1 30 4 127 1084 2 1.2500 5.0000 .25 .20
  8 1 31 4 219 1330 2 1.2500 5.0000 .25 .20
"),
+     header = TRUE)
> dunemp <- svydesign(id = ~id, weight = ~wt, data = unemp, fpc = rep(32,
+     8))
> psunemp <- postStratify(dunemp, ~poststr, data.frame(poststr = c(1,
+     2), Freq = c(7, 25)))
> svymean(~ue91, psunemp)

       mean     SE
ue91 565.81 187.93

> svytotal(~ue91, psunemp)

     total     SE
ue91 18106 6013.6

> svyby(~ue91, ~poststr, psunemp, svymean, keep.var = TRUE)

  poststr statistic.ue91        SE
1       1         1868.0 852.35238
2       2          201.2  30.07237

> svyby(~ue91, ~poststr, psunemp, svytotal, keep.var = TRUE)

  poststr statistic.ue91        SE
1       1          13076 5966.4666
2       2           5030  751.8092
```

5 PPS sampling

Sampling with probability-proportional-to-size, or other general forms of unequal probability sampling make full multistage analysis difficult. Standard error estimation requires either joint probabilities or conditional probabilities of sampling, but these depend on information about units not sampled. For this reason, most texts seem to resort to with-replacement analyses of PPS samples. R can handle these easily.

Here is an analysis of data from p350 of Levy and Lemeshow. The traditional analysis replicates the results from SUDAAN (`http://www.ats.ucla.edu/stat/sudaan/faq/svy_sudaan_pps.htm` and the replicate-weight analysis gives the same results as WesVar (`http://www.ats.ucla.edu/stat/wesvar/faq/pps/svy_wv_pps.htm`)

```
> hosp <- read.dta("hospslct.dta")
> dhosp <- svydesign(id = ~drawing, weight = ~wstar, data = hosp)
> svymean(~lifethrt + dxdead, dhosp)

         mean     SE
lifethrt 0.12 0.0200
dxdead   0.04 0.0245

> svytotal(~lifethrt + dxdead, dhosp)

          total     SE
lifethrt 6006.7 1001.1
dxdead   2002.2 1226.1

> svyratio(~dxdead, ~lifethrt, dhosp)

Ratio estimator: svyratio.survey.design2(~dxdead, ~lifethrt, dhosp)
Ratios=
        lifethrt
dxdead 0.3333333
SEs=
        lifethrt
dxdead 0.2324056

> rhosp <- as.svrepdesign(dhosp)
> svymean(~lifethrt + dxdead, rhosp)

         mean     SE
lifethrt 0.12 0.0200
dxdead   0.04 0.0245

> svytotal(~lifethrt + dxdead, rhosp)

          total     SE
lifethrt 6006.7 1001.1
dxdead   2002.2 1226.1
```

```
> svyratio(~dxdead, ~lifethrt, rhosp)

Ratio estimator: svyratio.svyrep.design(~dxdead, ~lifethrt, rhosp)
Ratios=
        lifethrt
dxdead 0.3333333
SEs=
     [,1]
[1,] 0.24
```