ch05output

2024-03-27

```
##### Install the R packages needed for the chapter

library(survey)
```

```
## Loading required package: grid

## Loading required package: Matrix

## Loading required package: survival

##
## Attaching package: 'survey'

## The following object is masked from 'package:graphics':
##
##     dotchart
```

```
library(sampling)
```

```
##
## Attaching package: 'sampling'

## The following objects are masked from 'package:survival':
##
##     cluster, strata
```

```
library(SDAResources)

########## Estimates from One-Stage Cluster Samples ##########

##### Example 5.2

data(gpa)
gpa
```

```
##    suite  gpa wt
## 1      1 3.08 20
## 2      1 2.60 20
## 3      1 3.44 20
## 4      1 3.04 20
## 5      2 2.36 20
```

```
## 6      2 3.04 20
## 7      2 3.28 20
## 8      2 2.68 20
## 9      3 2.00 20
## 10     3 2.56 20
## 11     3 2.52 20
## 12     3 1.88 20
## 13     4 3.00 20
## 14     4 2.88 20
## 15     4 3.44 20
## 16     4 3.64 20
## 17     5 2.68 20
## 18     5 1.92 20
## 19     5 3.28 20
## 20     5 3.20 20
```

```
wt<-rep(5,20)
# define one-stage cluster design
# note that id is suite instead of individual student as we take an SRS of suites
dgpa<-svydesign(id=~suite,weights=~wt,fpc=~rep(100,20),data=gpa)
dgpa
```

```
## 1 - level Cluster Sampling design
## With (5) clusters.
## svydesign(id = ~suite, weights = ~wt, fpc = ~rep(100, 20), data = gpa)
```

```
# estimate mean and se
gpamean<-svymean(~gpa,dgpa)
gpamean
```

```
##      mean     SE
## gpa 2.826 0.1637
```

```
degf(dgpa)
```

```
## [1] 4
```

```
# n=5, t-approximation is suggested for CI
confint(gpamean,level=.95,df=4) # use t-approximation
```

```
##        2.5 %   97.5 %
## gpa 2.371593 3.280407
```

```
# confint(gpamean,level=.95) # uses normal approximation, if desired (for large n)
# estimate total and se (if desired)
gpatotal<-svytotal(~gpa,dgpa)
gpatotal
```

```
##      total     SE
## gpa 1130.4 65.466
```

```
confint(gpatotal,level=.95,df=4)
```

```
##        2.5 %   97.5 %
## gpa 948.6374 1312.163
```

```
# you can also calculate SEs by direct formula
suitesum<-tapply(gpa$gpa,gpa$suite,sum)  #sum gpa for each suite
suitesum
```

```
##     1     2     3     4     5
## 12.16 11.36  8.96 12.96 11.08
```

```
# variability comes from among the suites
st2<-var(suitesum)
st2
```

```
## [1] 2.25568
```

```
# SE of t-hat, formula (5.3) of SDA
vthat <-100^2*(1-5/100)*st2/5
sqrt(vthat)
```

```
## [1] 65.46596
```

```
# SE of ybar, formula (5.6) of SDA
sqrt(vthat)/(4*100)
```

```
## [1] 0.1636649
```

```
##### Example 5.6

data(algebra)
nrow(algebra) #299
```

```
## [1] 299
```

```
head(algebra)
```

```
##   class Mi score
## 1    23 20    57
## 2    23 20    90
## 3    23 20    56
## 4    23 20    57
## 5    23 20    46
## 6    23 20    55
```

```r
algebra$sampwt<-rep(187/12,299)
# define one-stage cluster design
dalg<-svydesign(id=~class,weights=~sampwt,fpc=~rep(187,299), data=algebra)
dalg
```

```
## 1 - level Cluster Sampling design
## With (12) clusters.
## svydesign(id = ~class, weights = ~sampwt, fpc = ~rep(187, 299),
##     data = algebra)
```

```r
# estimate mean and se
svymean(~score,dalg)
```

```
##         mean     SE
## score 62.569 1.4916
```

```r
# n=12, t-distribution is suggested for CI
degf(dalg)
```

```
## [1] 11
```

```r
confint(svymean(~score,dalg),level=.95,df=11) #use t-approximation
```

```
##          2.5 %  97.5 %
## score 59.28562 65.8515
```

```r
# estimate total and se if desired
svytotal(~score,dalg)
```

```
##        total    SE
## score 291533 19893
```

```r
confint(svytotal(~score,dalg),level=.95,df=11)
```

```
##          2.5 %   97.5 %
## score 247749.4 335316.6
```

```r
########## Estimates from Multi-Stage Cluster Samples ##########

##### Example 5.8

data(coots)
# Want to estimate the mean egg volume
nrow(coots) #368
```

```
## [1] 368
```

```
head(coots)
```

```
##   clutch csize length breadth    volume tmt
## 1      1    13  44.30   31.10 3.7957569   1
## 2      1    13  45.90   32.70 3.9328497   1
## 3      2    13  49.20   34.40 4.2156036   1
## 4      2    13  48.70   32.70 4.1727621   1
## 5      3     6  51.05   34.25 0.9317646   0
## 6      3     6  49.35   34.40 0.9007362   0
```

```
coots$ssu<-rep(1:2,184) # index of ssu
coots$relwt<-coots$csize/2
head(coots)
```

```
##   clutch csize length breadth    volume tmt ssu relwt
## 1      1    13  44.30   31.10 3.7957569   1   1   6.5
## 2      1    13  45.90   32.70 3.9328497   1   2   6.5
## 3      2    13  49.20   34.40 4.2156036   1   1   6.5
## 4      2    13  48.70   32.70 4.1727621   1   2   6.5
## 5      3     6  51.05   34.25 0.9317646   0   1   3.0
## 6      3     6  49.35   34.40 0.9007362   0   2   3.0
```

```
attach(coots)
## with replacement variance, ignore fpc and second term
dcoots<-svydesign(id=~clutch+ssu,weights=~relwt,data=coots)
dcoots
```

```
## 2 - level Cluster Sampling design (with replacement)
## With (184, 368) clusters.
## svydesign(id = ~clutch + ssu, weights = ~relwt, data = coots)
```

```
# how about no weights included?
svydesign(id=~clutch+ssu,data=coots)
```

```
## Warning in svydesign.default(id = ~clutch + ssu, data = coots): No weights or
## probabilities supplied, assuming equal probability

## 2 - level Cluster Sampling design (with replacement)
## With (184, 368) clusters.
## svydesign(id = ~clutch + ssu, data = coots)
```

```
svymean(~volume,dcoots)  #ratio estimator
```

```
##          mean    SE
## volume 2.4908 0.061
```

```
confint(svymean(~volume,dcoots),level=.95,df=183)
```

```
##           2.5 %   97.5 %
## volume 2.370423 2.611134
```

```r
# now only include psu information, results are the same
dcoots2<-svydesign(id=~clutch,weights=~relwt,data=coots)
dcoots2
```

```
## 1 - level Cluster Sampling design (with replacement)
## With (184) clusters.
## svydesign(id = ~clutch, weights = ~relwt, data = coots)
```

```r
svymean(~volume,dcoots2)
```

```
##          mean    SE
## volume 2.4908 0.061
```

```r
##calculate using direct formulas

#split data using clutch, response interested is volume, list the volume values
#from the same clutch
coots2<-split(volume,clutch)
head(coots2)
```

```
## $`1`
## [1] 3.795757 3.932850
##
## $`2`
## [1] 4.215604 4.172762
##
## $`3`
## [1] 0.9317646 0.9007362
##
## $`4`
## [1] 3.018272 2.978397
##
## $`5`
## [1] 2.504580 2.486835
##
## $`6`
## [1] 3.971407 3.997112
```

```r
coots2[[1]]
```

```
## [1] 3.795757 3.932850
```

```r
nclus<-length(coots2) # number of clusters selected
nclus  #184
```

```
## [1] 184
```

```r
clusmean<-sapply(coots2,mean) #this is yibar
clusmean
```

```
##         1         2         3         4         5         6         7         8
## 3.8643033 4.1941828 0.9162504 2.9983346 2.4957075 3.9842595 1.9270690 2.9615264
##         9        10        11        12        13        14        15        16
## 3.4605792 2.9615264 3.4989084 2.9998683 3.5664408 2.9860652 2.9829979 2.4069825
##        17        18        19        20        21        22        23        24
## 2.0102296 2.4374025 2.9262519 2.9477233 2.5375350 4.2691555 3.7653876 2.5641525
##        25        26        27        28        29        30        31        32
## 1.9619759 3.4824816 2.5742925 1.9558159 3.4076484 2.9553917 2.5045800 1.5599376
##        33        34        35        36        37        38        39        40
## 2.4919050 2.4868350 1.9866161 1.9250156 1.4974752 1.9342557 1.5648048 2.4133200
##        41        42        43        44        45        46        47        48
## 2.8188946 1.2340760 1.2607823 3.0305418 2.4716250 2.0759368 3.1593705 1.9342557
##        49        50        51        52        53        54        55        56
## 1.9147489 1.5923856 2.4703575 3.0428112 2.0615634 2.4259950 1.5769728 1.9014021
##        57        58        59        60        61        62        63        64
## 2.6364000 0.8779212 0.8783775 0.9025614 1.4942304 1.5972528 1.4974752 2.0420566
##        65        66        67        68        69        70        71        72
## 0.8135829 1.2042644 0.6600506 2.0646434 1.5477696 3.5135100 2.5235925 1.9784027
##        73        74        75        76        77        78        79        80
## 2.0974970 1.5096432 3.9169983 2.0892836 2.4792300 1.5858960 3.0357563 1.5785952
##        81        82        83        84        85        86        87        88
## 1.2173070 1.1887375 1.1539574 0.5774730 0.6492769 2.1167985 2.4779625 2.3482947
##        89        90        91        92        93        94        95        96
## 3.0152051 1.9843574 2.0340485 2.4140805 1.1835205 2.5456470 1.2266231 0.8803852
##        97        98        99       100       101       102       103       104
## 0.5941406 1.9786081 2.9879056 2.3669295 3.4302809 0.6167655 1.1348282 1.1200467
##       105       106       107       108       109       110       111       112
## 1.9874375 3.0118310 2.0927743 1.9597172 2.4110385 1.9319970 2.6004030 3.6284976
##       113       114       115       116       117       118       119       120
## 1.6486829 1.6477094 2.8701194 2.1182359 2.5818975 1.6056893 3.9602683 3.1550762
##       121       122       123       124       125       126       127       128
## 2.0642328 3.6525902 2.5998960 2.9299327 3.0572277 2.5213110 3.4970832 2.9851450
##       129       130       131       132       133       134       135       136
## 1.9324077 2.1535535 3.5416181 2.6528775 2.9106084 1.5865450 2.8753339 2.0977024
##       137       138       139       140       141       142       143       144
## 2.3616060 2.6237250 1.5711322 2.0467793 2.8725733 2.8845359 2.9385213 3.7318039
##       145       146       147       148       149       150       151       152
## 2.5793625 2.6102895 2.0381552 2.6080080 3.1047717 2.9259452 1.1738318 1.5717811
##       153       154       155       156       157       158       159       160
## 1.9714213 2.4262485 1.8248121 2.5633920 2.3400585 1.9991416 1.7527396 2.9903595
##       161       162       163       164       165       166       167       168
## 1.9424691 1.5578285 1.5970906 1.1814089 2.4366420 1.6519277 1.6050403 0.8824842
##       169       170       171       172       173       174       175       176
## 2.3854350 2.4830325 2.5147200 2.9814642 2.3461425 3.6193716 2.4336000 3.7526112
##       177       178       179       180       181       182       183       184
## 2.5273950 3.1010909 1.9404158 1.9465758 3.4532784 4.2198877 4.4148166 3.4843068
```

```
clusvar<-sapply(coots2,var) #this is si
clusvar
```

```
##            1            2            3            4            5            6
## 9.397218e-03 9.176971e-04 4.813808e-04 7.950297e-04 1.574425e-04 3.303709e-04
##            7            8            9           10           11           12
## 5.061604e-03 5.123002e-03 1.066034e-04 2.239726e-02 3.524574e-03 1.693554e-04
```

```
##           13           14           15           16           17           18
## 1.289901e-02 2.940199e-03 1.058472e-03 2.008195e-03 5.396795e-04 2.891801e-05
##           19           20           21           22           23           24
## 7.526909e-05 4.156735e-02 2.172064e-03 1.696822e-02 3.524574e-03 1.699737e-03
##           25           26           27           28           29           30
## 3.562728e-04 2.729046e-02 2.602621e-04 3.897920e-03 8.161820e-03 2.305116e-04
##           31           32           33           34           35           36
## 1.285245e-05 3.803503e-04 2.172064e-03 5.140980e-03 4.743277e-04 3.543755e-03
##           37           38           39           40           41           42
## 2.321584e-03 8.432492e-06 8.225568e-04 8.225568e-04 2.049201e-02 2.229543e-04
##           43           44           45           46           47           48
## 5.215126e-04 3.010764e-04 1.285245e-03 1.020332e-03 0.000000e+00 2.436990e-03
##           49           50           51           52           53           54
## 1.897311e-05 1.592470e-04 2.602621e-04 9.954337e-03 8.432492e-06 0.000000e+00
##           55           56           57           58           59           60
## 1.316091e-04 3.035697e-04 5.140980e-05 1.349199e-04 9.369436e-05 2.814995e-04
##           61           62           63           64           65           66
## 1.521401e-03 1.316091e-06 6.369880e-04 1.115197e-03 3.372997e-05 8.401290e-04
##           67           68           69           70           71           72
## 2.429916e-05 2.025906e-03 1.347677e-03 3.264728e-04 4.887144e-03 5.483228e-03
##           73           74           75           76           77           78
## 1.115197e-03 6.962121e-04 1.798686e-05 9.296823e-04 8.225568e-04 1.184482e-05
##           79           80           81           82           83           84
## 9.220463e-06 3.837721e-03 1.493563e-03 1.234349e-05 1.234349e-05 3.936063e-05
##           85           86           87           88           89           90
## 1.255122e-04 8.264686e-04 3.499080e-03 4.866865e-01 3.010764e-04 8.634872e-03
##           91           92           93           94           95           96
## 5.526318e-03 1.641129e-03 1.186210e-04 1.390121e-03 2.595219e-05 6.464328e-04
##           97           98           99          100          101          102
## 5.020488e-06 2.732128e-05 1.881727e-07 5.296366e-03 4.480006e-04 3.277696e-04
##          103          104          105          106          107          108
## 6.816694e-03 1.590120e-03 2.550829e-04 4.066224e-03 3.372997e-03 1.174140e-03
##          109          110          111          112          113          114
## 4.210591e-03 3.543755e-03 8.688256e-05 1.539353e-03 2.547952e-05 8.625133e-04
##          115          116          117          118          119          120
## 9.220463e-06 3.035697e-04 9.728019e-04 5.732892e-05 1.321484e-03 1.693554e-04
##          121          122          123          124          125          126
## 6.092476e-04 2.665084e-03 1.499110e-03 2.709687e-03 2.576085e-04 1.234349e-03
##          127          128          129          130          131          132
## 6.822615e-05 1.012821e-02 1.076745e-03 3.899185e-04 3.080837e-04 8.432492e-04
##          133          134          135          136          137          138
## 6.164726e-03 1.478760e-04 2.438718e-04 3.372997e-07 8.225568e-06 3.048087e-03
##          139          140          141          142          143          144
## 5.264364e-04 1.057772e-03 2.575896e-03 6.774218e-04 5.487117e-04 9.277158e-04
##          145          146          147          148          149          150
## 1.041048e-05 1.291530e-02 1.487492e-04 4.940482e-04 2.114309e-03 1.166690e-02
##          151          152          153          154          155          156
## 3.159934e-05 4.851637e-04 1.417502e-04 6.622739e-03 3.979875e-02 1.235120e-02
##          157          158          159          160          161          162
## 2.343374e-02 8.103625e-03 5.186320e-03 1.867144e-02 1.296580e-03 5.732892e-03
##          163          164          165          166          167          168
## 3.343081e-03 3.733907e-06 4.751088e-02 1.770932e-04 1.900435e-05 1.125998e-05
##          169          170          171          172          173          174
## 2.056392e-04 6.506553e-03 8.688256e-03 1.693554e-04 3.213112e-06 7.255691e-03
```

```
##          175          176          177          178          179          180
## 8.225568e-04 2.665084e-03 3.213112e-04 1.272048e-02 1.425091e-03 7.589243e-05
##          181          182          183          184
## 1.705654e-03 3.670788e-05 8.819069e-03 6.662710e-06
```

```
#split data using clutch, response interested is csize
coots3<-split(csize,clutch)
head(coots3)
```

```
## $`1`
## [1] 13 13
##
## $`2`
## [1] 13 13
##
## $`3`
## [1] 6 6
##
## $`4`
## [1] 11 11
##
## $`5`
## [1] 10 10
##
## $`6`
## [1] 13 13
```

```
mi<-sapply(coots3,length)
mi
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
## 181 182 183 184
##   2   2   2   2
```

```
Mi<-sapply(coots3,mean)
Mi
```

```
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
##  13  13   6  11  10  13   9  11  12  11  12  11  12  11  11  10   9  10  11  11
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40
##  10  13  12  10   9  12  10   9  12  11  10   8  10  10   9   9   8   9   8  10
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60
##  11   7   7  11  10   9  11   9   9   8  10  11   9  10   8   9  10   6   6   6
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80
##   8   8   8   9   6   7   5   9   8  12  10   9   9   8  13   9  10   8  11   8
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99 100
##   7   7   7   5   5   9  10  10  11   9   9  10   7  10   7   6   5   9  11  10
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120
##  12   5   7   7   9  11   9   9  10   9  10  12   8   8  11   9  10   8  13  11
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140
##   9  12  10  11  11  10  12  11   9   9  12  10  11   8  11   9  10  10   8   9
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160
##  11  11  11  12  10  10   9  10  11  11   7   8   9  10   9  10  10   9   9  11
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
##   9   8   8   7  10   8   8   6  10  10  10  11  10  12  10  12  10  11   9   9
## 181 182 183 184
##  12  13  13  12
```

```
##unbiased estimator, suppose N is known as 10000, 1-n/N goes to 1
##suppose M0=95000
N<-10000
tunb<-N*sum(Mi*clusmean)/nclus
tunb
```

```
## [1] 237950.8
```

```
yunb<-tunb/95000
yunb
```

```
## [1] 2.504745
```

```
#variances
ssufpc<-1-mi/Mi
head(ssufpc)
```

```
##         1         2         3         4         5         6
## 0.8461538 0.8461538 0.6666667 0.8181818 0.8000000 0.8461538
```

```
st2<-var(Mi*clusmean)
st2
```

```
## [1] 149.5268
```

```
vartunb<-N^2*(1-nclus/N)*st2/nclus + (N/nclus)*sum(ssufpc*(Mi^2)*clusvar/mi)
vartunb
```

```
## [1] 79771832
```

```
varyunb<-vartunb/(95000)^2
varyunb
```

```
## [1] 0.008838984
```

```
seyunb<-sqrt(varyunb)
seyunb
```

```
## [1] 0.09401587
```

```
##ratio estimation
ybarratio<-sum(Mi*clusmean)/sum(Mi)
ybarratio
```

```
## [1] 2.490498
```

```
sr2<-var(Mi*(clusmean - ybarratio))
varybarr<-( (1 - nclus/N)*sr2/nclus + sum(ssufpc*(Mi^2)*clusvar/mi)/(nclus*N) )/(mean(Mi))^2
varybarr
```

```
## [1] 0.003653726
```

```
seybarr<-sqrt(varybarr)
seybarr
```

```
## [1] 0.06044606
```

```
#compare first term and second term in variance formula
vterm1<-((1 - nclus/N)*sr2/nclus)/(mean(Mi))^2
vterm2<- (sum(ssufpc*(Mi^2)*clusvar/mi)/(nclus*N))/(mean(Mi))^2
vterm1
```

```
## [1] 0.00365345
```

```
vterm2
```

```
## [1] 2.757192e-07
```

```
#textbook considers N goest to infinity, use ssufpc<-1, and ignore second term
#same as first stage with replacement variance
varybarrtext<-(sr2/nclus)/(mean(Mi))^2
varybarrtext
```

```
## [1] 0.003721934
```

```r
seybarrtext<-sqrt(varybarrtext)
seybarrtext
```

```
## [1] 0.06100765
```

```r
list(yunb=yunb,seyunb=seyunb,yratio=ybarratio,seybarr<-seybarr,seybarrtext=seybarrtext)
```

```
## $yunb
## [1] 2.504745
##
## $seyunb
## [1] 0.09401587
##
## $yratio
## [1] 2.490498
##
## [[4]]
## [1] 0.06044606
##
## $seybarrtext
## [1] 0.06100765
```

```r
##use svymean and svytotal to calculate the estimates
cootsfpc1<-rep(10000,368) #Note: 10000 is an arbitrary number, we don't have information of N
cootsfpc2<-coots$csize #number of ssus in each psu
ssu<-rep(1:2,184) #index of ssu
dcoots3<-svydesign(id=~clutch+ssu,fpc=~cootsfpc1+cootsfpc2,data=coots)
```

```
## Warning in as.fpc(fpc, strata, ids, pps = pps): 'fpc' varies within strata:
## stratum 1.88 at stage 2
```

```r
dcoots3
```

```
## 2 - level Cluster Sampling design
## With (184, 368) clusters.
## svydesign(id = ~clutch + ssu, fpc = ~cootsfpc1 + cootsfpc2, data = coots)
```

```r
svymean(~volume,dcoots3)  #ratio estimator
```

```
##          mean     SE
## volume 2.4908 0.0604
```

```r
svytotal(~volume,dcoots3)  #unbiased estimator of the total
```

```
##         total     SE
## volume 237978 8931.6
```

```
##### Example 5.7

### With-replacement variance

data(schools)
head(schools)
```

```
##   schoolid gender math reading mathlevel readlevel  Mi finalwt
## 1        9      F   42      42         2         2 163  61.125
## 2        9      F   29      30         1         1 163  61.125
## 3        9      M   31      25         1         1 163  61.125
## 4        9      F   22      33         1         2 163  61.125
## 5        9      M   35      36         1         2 163  61.125
## 6        9      F   30      17         1         1 163  61.125
```

```
unique(schools$schoolid)  #10 schools
```

```
##  [1]  9 17 18 22 35 43 46 55 62 75
```

```
schools$finalwt[1:30]
```

```
##  [1] 61.125 61.125 61.125 61.125 61.125 61.125 61.125 61.125 61.125 61.125
## [11] 61.125 61.125 61.125 61.125 61.125 61.125 61.125 61.125 61.125 61.125
## [21] 67.500 67.500 67.500 67.500 67.500 67.500 67.500 67.500 67.500 67.500
```

```
# calculate with-replacement variance; no fpc argument
# include psu variable in id; include weights
dschools<-svydesign(id=~schoolid,weights=~finalwt,data=schools)
# dschools tells you this is treated as a with-replacement sample
dschools
```

```
## 1 - level Cluster Sampling design (with replacement)
## With (10) clusters.
## svydesign(id = ~schoolid, weights = ~finalwt, data = schools)
```

```
mathmean<-svymean(~math,dschools)
mathmean
```

```
##        mean     SE
## math 33.123 1.7599
```

```
degf(dschools)
```

```
## [1] 9
```

```
# use t distribution for confidence intervals because there are only 10 psus
confint(mathmean,df=degf(dschools))
```

```
##         2.5 %  97.5 %
## math 29.14179 37.1041
```

```
# estimate proportion and total number of students with mathlevel=2
svymean(~factor(mathlevel),dschools)
```

```
##                         mean     SE
## factor(mathlevel)1 0.71231 0.0542
## factor(mathlevel)2 0.28769 0.0542
```

```
svytotal(~factor(mathlevel),dschools)
```

```
##                        total      SE
## factor(mathlevel)1 12303.4 2244.14
## factor(mathlevel)2  4969.1  676.26
```

```
### Without-replacement variance

# create a variable giving each student an id number
schools$studentid<-1:(nrow(schools))
head(schools)
```

```
##   schoolid gender math reading mathlevel readlevel  Mi finalwt studentid
## 1        9      F   42      42         2         2 163  61.125         1
## 2        9      F   29      30         1         1 163  61.125         2
## 3        9      M   31      25         1         1 163  61.125         3
## 4        9      F   22      33         1         2 163  61.125         4
## 5        9      M   35      36         1         2 163  61.125         5
## 6        9      F   30      17         1         1 163  61.125         6
```

```
# calculate without-replacement variance
# specify both stages of the sample in the id argument
# give both sets of population sizes in the fpc argument
# do not include the weight argument
dschoolwor<-svydesign(id=~schoolid+studentid,fpc=~rep(75,nrow(schools))+Mi,
                      data=schools)
dschoolwor
```

```
## 2 - level Cluster Sampling design
## With (10, 200) clusters.
## svydesign(id = ~schoolid + studentid, fpc = ~rep(75, nrow(schools)) +
##     Mi, data = schools)
```

```
mathmeanwor<-svymean(~math,dschoolwor)
mathmeanwor
```

```
##        mean     SE
## math 33.123 1.6605
```

```
confint(mathmeanwor,df=degf(dschoolwor))
```

```
##         2.5 %   97.5 %
## math 29.36667 36.87923
```

```r
# estimate proportion and total number of students with mathlevel=2
svymean(~factor(mathlevel),dschoolwor)
```

```
##                         mean     SE
## factor(mathlevel)1 0.71231 0.0516
## factor(mathlevel)2 0.28769 0.0516
```

```r
svytotal(~factor(mathlevel),dschoolwor)
```

```
##                        total      SE
## factor(mathlevel)1 12303.4 2097.83
## factor(mathlevel)2  4969.1  657.69
```