ch02output

```
# Code from Chapter 2 of R Companion for Sampling: Design and Analysis by Yan Lu
# and Sharon L. Lohr
# All code is presented for educational purposes only and without warranty.

##### 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)

########## Selecting a Simple Random Sample ##########

##### Example 2.5

# Set the seed for random number generation
set.seed(108742)
# Select an SRS of size n=4 from a population of size N=10 without replacement
srs4 <- sample(1:10, 4, replace = FALSE)
# Print the sample to see
# Can print an object by typing its name, or typing "print(object)"
srs4
```

```
## [1] 1 8 9 5
```

```
# Run again, without setting a new seed.
sample(1:10, 4, replace = FALSE)
```

```
## [1]  9  7  3 10
```

```r
# Now go back to original seed.
set.seed(108742)
sample(1:10, 4, replace = FALSE)
```

```
## [1] 1 8 9 5
```

```r
# Using the sample function to select an SRS with replacement
set.seed(101)
srswr4 <- sample(1:10, 4, replace = TRUE)
srswr4
```

```
## [1] 9 9 7 1
```

```r
# Using function srswor from the sampling package

set.seed(1329)
# Select an SRS of size n=4 from a population of size N=10 without replacement.
s1<-srswor(4,10)
# List the units in the sample (the population units having s1=1).
s1
```

```
##  [1] 0 0 1 1 1 0 0 0 1 0
```

```r
(1:10)[s1==1]
```

```
## [1] 3 4 5 9
```

```r
# Select an SRS of size n=4 from a population of size N=10 with replacement.
set.seed(35882)
s2<-srswr(4,10)
# the selected units are 2 and 9
s2
```

```
##  [1] 0 2 0 0 0 0 0 0 2 0
```

```r
(1:10)[s2!=0]
```

```
## [1] 2 9
```

```r
# number of replicates, units 2 and 9 both appear twice
s2[s2!=0]
```

```
## [1] 2 2
```

```r
# can use the getdata function to extract the sample from data frame with population
popdf<-data.frame(popid=1:10)
getdata(popdf,s2)
```

```
## [1] 2 2 9 9
```

```r
# Call a function using variable names
set.seed(35882)
srswr(4,10)
```

```
##  [1] 0 2 0 0 0 0 0 0 2 0
```

```r
set.seed(35882)
srswr(n=4,N=10)
```

```
##  [1] 0 2 0 0 0 0 0 0 2 0
```

```r
set.seed(35882)
srswr(N=10,n=4)
```

```
##  [1] 0 2 0 0 0 0 0 0 2 0
```

```r
##### Example 2.6, select the sample

# Select a different sample of size 300 from agpop
# Load the SDAResources package containing all the data in SDA book
# library(SDAResources)  # we comment this since we already loaded the package
data(agpop)  # Load the data set agpop
N <- nrow(agpop)
N  # 3078 observations
```

```
## [1] 3078
```

```r
# Select an SRS of size n=300 from agpop
set.seed(8126834)
index <- srswor(300,N)
# each unit k is associated with index 1 or 0, with 1 indicating selection
index[1:10]
```

```
##  [1] 0 0 0 1 0 0 0 0 0 0
```

```r
# agsrs2 is an SRS with size 300 selected from agpop
# extract the sampled units from the data frame containing the population
agsrs2 <- getdata(agpop,index)
agsrs2 <- agpop[(1:N)[index==1],] # alternative way to extract the sampled units
head(agsrs2)
```

```
##            county state acres92 acres87 acres82 farms92 farms87 farms82
## 4     JUNEAU AREA    AK     210     214     127       8       8      12
## 30 DE KALB COUNTY    AL  210733  213440  221502    1894    2047    2228
## 38    HALE COUNTY    AL  167583  154581  179618     382     441     481
## 46     LEE COUNTY    AL   67962   79836  100949     336     402     407
## 50 MADISON COUNTY    AL  224370  235478  292873     871     977    1101
## 62 RUSSELL COUNTY    AL  112620  143568  141048     213     276     314
##    largef92 largef87 largef82 smallf92 smallf87 smallf82 region
## 4         0        0        0        5        4        8      W
## 30       13        5        6      114      133      168      S
## 38       38       33       39       12       22       17      S
## 46       10       10       20       15       22       20      S
## 50       59       59       61       46       76       89      S
## 62       25       30       33       14       14       25      S
```

```r
# Create the variable of sampling weights
n <- nrow(agsrs2)
agsrs2$sampwt <- rep(3078/n,n)
# Check that the weights sum to N
sum(agsrs2$sampwt)
```

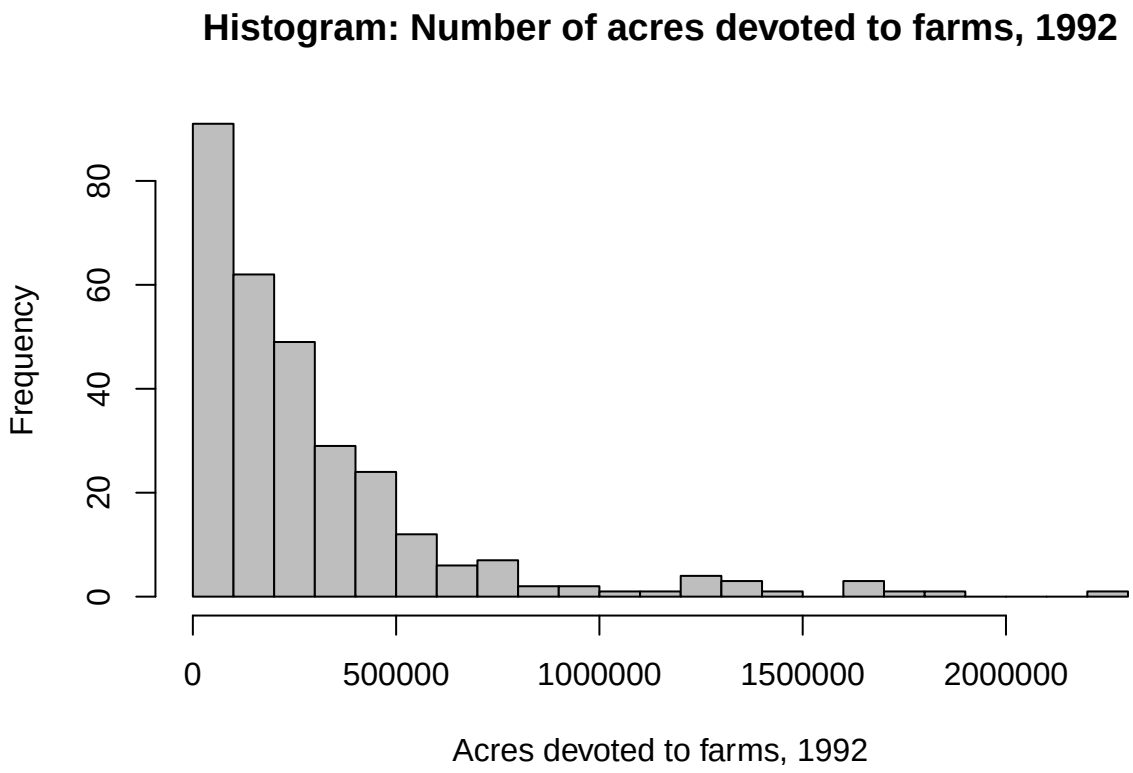
```
## [1] 3078
```

```r
########## Computing Statistics from a Simple Random Sample ##########

##### Examples 2.6, 2.7, and 2.11

# Draw a histogram
```

```r
hist(agsrs$acres92,breaks=20,col="gray",xlab="Acres devoted to farms, 1992",
     main="Histogram: Number of acres devoted to farms, 1992")
```




```r
# Base R functions such as t.test will calculate statistics for an SRS,
# but without the fpc.
t.test(agsrs$acres92)
```

```
##
##  One Sample t-test
##
## data:  agsrs$acres92
## t = 14.975, df = 299, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  258749.6 337044.5
## sample estimates:
## mean of x
##    297897
```

```r
# Calculate the statistics by direct formula
n <- length(agsrs$acres92)
ybar <- mean(agsrs$acres92)
ybar
```

```
## [1] 297897
```

```r
hatvybar<-(1-n/3078)*var(agsrs$acres92)/n
seybar<-sqrt(hatvybar)
seybar
```

```
## [1] 18898.43
```

```r
# Calculate confidence interval by direct formula using t distribution
Mean_CI <- c(ybar - qt(.975, n-1)*seybar, ybar + qt(.975, n-1)*seybar)
names(Mean_CI) <- c("lower", "upper")
Mean_CI
```

```
##    lower    upper
## 260706.3 335087.8
```

```r
# To obtain estimates for the population total,
# multiply each of ybar, seybar, and Mean_CI by N = 3078
seybar*3078
```

```
## [1] 58169381
```

```r
Mean_CI*3078
```

```
##      lower      upper
##  802453859 1031400361
```

```r
# Calculate coefficient of variation of mean
seybar/ybar
```

```
## [1] 0.06343948
```

```r
### Using functions in the survey package

# Create the variable of sampling weights
n <- nrow(agsrs)
agsrs$sampwt <- rep(3078/n,n)
# Create variable lt200k
agsrs$lt200k <- rep(0,n)
agsrs$lt200k[agsrs$acres92 < 200000] <- 1
# look at the first 10 observations with column 3 (acres92) and column 17 (lt200k)
agsrs[1:10,c(3,17)]
```

```
##    acres92 lt200k
## 1   175209      1
## 2   138135      1
## 3    56102      1
## 4   199117      1
## 5    89228      1
## 6    96194      1
## 7    57253      1
## 8   210692      0
## 9    78498      1
## 10  219444      0
```

```r
# Specify the survey design.
# This is an SRS, so the only design features needed are the weights
#    or information used to calculate the fpc.
dsrs <- svydesign(id = ~1, weights = ~sampwt, fpc = rep(3078,300), data = agsrs)
dsrs
```

```
## Independent Sampling design
## svydesign(id = ~1, weights = ~sampwt, fpc = rep(3078, 300), data = agsrs)
```

```
# Calculate the mean for acres92 and its standard error using the svymean function.
smean <- svymean(~acres92,dsrs)
smean
```

```
##           mean    SE
## acres92 297897 18898
```

```
# Use the confint function to compute a 95% confidence interval from
# the information in smean, df = n-1 = 300-1 = 299
confint(smean, df=299)
```

```
##            2.5 %   97.5 %
## acres92 260706.3 335087.8
```

```
# Repeat these steps with the svytotal function to obtain estimated totals.
stotal <- svytotal(~acres92,dsrs)
stotal
```

```
##             total       SE
## acres92 916927110 58169381
```

```
confint(stotal, df=299)
```

```
##             2.5 %     97.5 %
## acres92 802453859 1031400361
```

```
# Calculate the CV of the mean
SE(smean)/coef(smean)
```

```
##            acres92
## acres92 0.06343948
```

```
# or
smean<-as.data.frame(smean)
smean[[2]]/smean[[1]]
```

```
## [1] 0.06343948
```

```
# Estimate population means for multiple variables
agsrs_means <- svymean(~acres92+lt200k,dsrs)
agsrs_means
```

```
##              mean         SE
## acres92 297897.05 18898.4344
## lt200k       0.51     0.0275
```

```
confint(agsrs_means, df=299)
```

```
##                2.5 %       97.5 %
## acres92 2.607063e+05 3.350878e+05
## lt200k  4.559508e-01 5.640492e-01
```

```
# Analyzing a categorical variable that is coded as characters
# First, display the category names and counts
table(agsrs$region)
```

```
##
##  NC  NE   S   W
## 107  24 130  39
```

```r
# Find the estimated proportions in each category
region_prop <- svymean(~region,dsrs)
region_prop
```

```
##             mean     SE
## regionNC 0.35667 0.0263
## regionNE 0.08000 0.0149
## regionS  0.43333 0.0272
## regionW  0.13000 0.0185
```

```r
confint(region_prop,df=299)
```

```
##                2.5 %    97.5 %
## regionNC 0.30487557 0.4084578
## regionNE 0.05066780 0.1093322
## regionS  0.37975605 0.4869106
## regionW  0.09363889 0.1663611
```

```r
region_total <- svytotal(~region,dsrs)
region_total
```

```
##            total     SE
## regionNC 1097.82 81.005
## regionNE  246.24 45.878
## regionS  1333.80 83.799
## regionW   400.14 56.872
```

```r
confint(region_total,df=299)
```

```
##              2.5 %    97.5 %
## regionNC  938.4070 1257.2330
## regionNE  155.9555  336.5245
## regionS  1168.8891 1498.7109
## regionW   288.2205  512.0595
```

```r
# Analyzing a categorical variable that is coded as numbers
# First, analyze lt200k as a numeric variable (works only if all values are 0 or 1)
# This gives the mean of variable lt200k, which is the proportion with lt200k = 1.
svymean(~lt200k,dsrs)
```

```
##        mean     SE
## lt200k 0.51 0.0275
```

```r
# Now, analyze lt200k as a factor variable. This gives the proportion
# in each category.
svymean(~factor(lt200k),dsrs)
```

```
##                 mean     SE
## factor(lt200k)0 0.49 0.0275
## factor(lt200k)1 0.51 0.0275
```

```r
##### Exercise 2.27 of SDA

# Calculating bootstrap means for Exercise 2.27 in SDA
set.seed (244)
B = 1000
n = length(agsrs$acres92)
boot.samples = matrix(sample(agsrs$acres92, size = B * n, replace = TRUE),B, n)
```

```r
boot.statistics = apply(boot.samples, 1, mean)
hist(boot.statistics, main="Estimated Sampling Distribution of ybar",
     xlab="Mean of acres92 from Bootstrap Replicate",
     col="gray",border="white")
```