STAT 4234/5234 Survey Sampling: Introduction to R

Data sets in R are either vectors or matrices. We begin by creating a vector and doing some elementary operations on it.

```
> x <- c(1,2,4)
> x
[1] 1 2 4
> x[3]
[1] 4
> x[3] <- 5
> x
[1] 1 2 5
> y <- x + 4
> y
[1] 5 6 9
> z <- c(x,y)
> z
[1] 1 2 5 5 6 9
> x * y
[1]  5 12 45
```

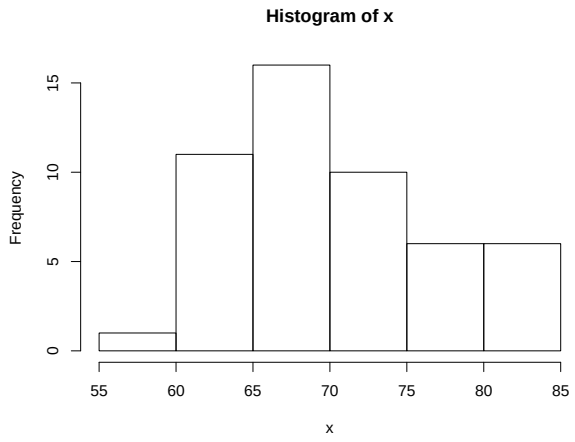
Next we generate a random sample of size 50 from a Normal population with mean 70 and standard deviation 6.

```
> x <- rnorm(50, mean=70, sd=6)
> x
 [1] 63.86253 56.65902 67.40514 60.51333 64.39782 60.53715 73.06585 61.85700
 [9] 64.29876 67.66324 67.35130 64.12132 63.83110 78.94751 74.41249 81.59720
[17] 80.24328 71.83290 69.61762 73.65558 80.59062 76.87083 64.59808 72.18889
[25] 66.16533 75.23247 67.51224 79.91131 66.67817 61.56949 81.33071 71.01274
[33] 68.88055 72.93181 82.27148 77.02583 64.64700 68.32732 68.99791 65.39971
[41] 68.78031 70.18405 68.33129 72.26535 65.01912 68.07404 66.90789 70.68199
[49] 77.27659 80.89803
> 1:5
[1] 1 2 3 4 5
> x[1:5]
[1] 63.86253 56.65902 67.40514 60.51333 64.39782
> c(1,3,5)
[1] 1 3 5
> x[c(1,3,5)]
[1] 63.86253 67.40514 64.39782
> sum(x)
[1] 3506.431
> length(x)
[1] 50
> sum(x)/length(x)
[1] 70.12863
```

```
> mean(x)
[1] 70.12863
> var(x)
[1] 41.14463
> sum( (x-mean(x))^2 ) / (length(x) - 1)
[1] 41.14463
> min(x)
[1] 56.65902
> max(x)
[1] 82.27148
> quantile(x)
      0%      25%      50%      75%     100%
56.65902 65.11427 68.83043 74.22326 82.27148
> quantile(x, .60)
     60%
70.81429
> quantile(x, .80)
     80%
76.90183
> quantile(x, c(.60, .80))
     60%      80%
70.81429 76.90183
```

You can make lots of nice plots in R, for example a histogram of this random sample of size 50 from the Normal distribution.

```
> hist(x)
```



The function `sample` allows one to take random samples from a vector. Pick 5 numbers at random from $1, 2, \ldots, 20$:

```
> sample(1:20, 5)
[1] 17 10  1  5 15
```

Enter `help(sample)` for information about how the function `sample` works.

A nice thing about R is that it is easy to write your own functions to compute quantities of interest. The following simple example extracts the last value and maximum value of a vector.

```
> last.and.max <- function(x)
+ {
+  n <- length(x)
+  ans1 <- x[n]
+  ans2 <- max(x)
+  answer <- c(last=ans1, max=ans2)
+  return(answer)
+ }
> x <- c(1,2,3,4,5,20,0)
> last.and.max(x)
last  max
   0   20
```

A less simple example: The following function allows you to take repeated random samples of size n from a population, and calculate the mean for each sample; then returns the vector of sample means.

```
> sample.means <- function(y, n, n.samples)
+ {
+  answer <- rep(NA, n.samples)
+  for(i in 1:n.samples)
+  {
+   samp <- sample(y, n)
+   answer[i] <- mean(samp)
+  }
+  return(answer)
+ }
```

For a population to test it on, let's consider a population of $N = 500$ draws from a gamma distribution with shape parameter $\alpha = 2$ and rate parameter $\lambda = 1$, having density

$$f(x) = \frac{\lambda^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\lambda x} \quad \text{for} \quad x > 0 \, .$$

```
> y <- rgamma(n=500, shape=2, rate=1)
> out <- sample.means(y=y, n=20, n.samples=50)
> out
 [1] 1.795636 1.716149 2.272027 1.820216 2.004564 2.300934 2.149115 2.170561
 [9] 2.573799 1.997914 2.799264 1.365858 1.913659 2.144246 2.646175 2.038747
[17] 2.522094 2.121987 1.770390 1.679328 1.764967 2.312405 1.723115 2.585771
[25] 2.111660 2.319779 1.666684 2.116039 1.990756 2.158514 2.268865 1.805439
[33] 1.714186 1.801261 2.074543 2.445966 1.685294 2.074893 2.525372 2.305015
[41] 2.630593 1.850693 1.756584 2.098937 2.131098 2.272648 2.049089 2.022983
[49] 2.161136 1.831304
> round(out, digits=2)
 [1] 1.80 1.72 2.27 1.82 2.00 2.30 2.15 2.17 2.57 2.00 2.80 1.37 1.91 2.14 2.65
[16] 2.04 2.52 2.12 1.77 1.68 1.76 2.31 1.72 2.59 2.11 2.32 1.67 2.12 1.99 2.16
[31] 2.27 1.81 1.71 1.80 2.07 2.45 1.69 2.07 2.53 2.31 2.63 1.85 1.76 2.10 2.13
[46] 2.27 2.05 2.02 2.16 1.83
```