Stat 2080: Statistical Modeling

Introduction to R: 2

Vectorized Computations

Some operators and functions are designed to work in an intuitive way when used with objects such as vectors. The operator or function is applied to each individual element of the object. This is called *vectorized computation* since the process is performed on entire vectors rather than one at a time.

```
> h <- c(15.1, 11.3, 7, 9)
> h^2
[1] 228.01 127.69  49.00  81.00
> 1/h
[1] 0.06622517 0.08849558 0.14285714 0.11111111
> log(h)
[1] 2.714695 2.424803 1.945910 2.197225
> sqrt(h)
[1] 3.885872 3.361547 2.645751 3.000000
> sum(h)
[1] 42.4
> sum(h^2)
[1] 485.7
```

Consider the following example where vectors of different lengths are used in an arithmetic expression:

```
> hh <- c(h, 0, 0, 0, 0, h)
> hh
 [1] 15.1 11.3  7.0  9.0  0.0  0.0  0.0  0.0 15.1 11.3  7.0  9.0
> hhh <- 2*h + hh + 1
> hhh
 [1] 46.3 34.9 22.0 28.0 31.2 23.6 15.0 19.0 46.3 34.9 22.0 28.0
```

In the expression 2*h + hh + 1, the operator 2*h produces a vector of length 4. Then it is added to hh, a vector of length 12. To perform this operation, the vector resulting from the operation 2*h is replicated three times to match the length of hh. The result is a vector of length 12, where each element has 1 added to it to produce the final output. This is only possible when the length of the longer vector is a multiple of the shorter vector. For example, the following would not work:

```
> hh <- c(h, 0, 0, 0, h)
> hh
 [1] 15.1 11.3  7.0  9.0  0.0  0.0  0.0 15.1 11.3  7.0  9.0
> hhh <- 2*h + hh + 1
Warning message:
```

```
In 2 * h + hh :
  longer object length is not a multiple of shorter object length
```

Exercise 1: Write code that will compute the following equation:

$$\sum_{x=1}^{5} \frac{\log(x)^2}{3x}$$

Logical Expressions

A logical expression is one that results in either a TRUE or FALSE statement. For example if you would like to ask R if 4 is less than 6, you would write:

```
> 4 < 6
[1] TRUE
```

Obviously this is a very simple example of how to use logical statements in R, but at its core it is only answering a true/false or yes/no question. Here is a list of the logical operators and their meaning in R:

Operator	Logical meaning	Example
==	Exactly equal to	`> 2 == 2` `[1] TRUE`
!=	Not equal to	`> 2 != 2` `[1] FALSE`
<	Less than	`> 1 < 2` `[1] TRUE`
<=	Less than or equal to	`> 2 <= 1` `[1] FALSE`
>	Greater than	`> 2 > 1` `[1] TRUE`
>=	Greater than or equal to	`> 1 >= 2` `[1] FALSE`
| or ||	Or	`> 1<2 || 2<1` `[1] TRUE`
& or &&	And	`> 1<2 && 2<1` `[1] FALSE`

Logical expressions can be applied to objects and vectors as well

```
> x <- 3.5
> y <- -5.7
> x<5 && y>-3
[1] FALSE
> h
[1] 15.1 11.3  7.0  9.0
> h > 9
[1]  TRUE  TRUE FALSE FALSE


> sum(h>9)
[1] 2
```

As seen in the above example, the comparison operator > is vectorized. It compares each value of the vector h to the constant 9 and produces a TRUE or FALSE value. When using arithmetic operations on logical outcomes, values of 1 and 0 are assigned to T and F, respectively. Thus, the expression sum(h<9) counts the number of values in h that are larger than nine and returns the total of two. Arithmetic operations will still take precedence over logical values, therefore in the example below, the addition is performed before the logical comparison.

```
> x + y < -3
[1] FALSE
```

Exercise 2: Write the code and answer the following logical questions using the values below.

$q = 6$ $w = -2.5$ $e = 0$ $r = (1,\ 2,\ 3,\ 4,\ 5)$ $t = (-0.5,\ 3,\ -4,\ 2.7, -6.1)$

Is *q* greater than or equal to *w*?

Is *e* less than *q* and *w*?

Is *w* greater than *e* or *q* greater than *w* or *e* greater then *q*?

How many values of $r + t$ are greater than 0?

Are there any terms in $r \cdot t$ or r/t that are greater than 11? If yes write code to count how many.

Built in R Functions

Most computations in R involve using functions that are already built into R. R contains a large number of built-in functions. Usually the functions will be used with an object and create an output in your console window. Some will produce a *side effect* such as a plot or graph. We have seen a few of the built-in functions already, such as sum(), log(), seq(), rep(), and sqrt() but there are many more.

Exercise 3: Find the R functions for other commonly used math and statistical applications.

Average: _____ Variance: _____

Standard Deviation: _____ Median: _____

5 number summary: _____ Factorial (!):_____

Product: _____ How long a vector is: _____

Hint: Even the best R programmers in the world do not know or remember every single function ever written. But they do know how to find them and figure out how to use them properly. When in doubt about whether there is a built-in R function to do what you want, try searching in the Help tab, or simply Googling "How to _____ in R". There is nothing wrong with Googling things, it's how the pros do it, knowing how to read the documentation on how to use new functions is where you will need to practice and build experience.

Using a preloaded data set in R called *chickwts* lets consider how to calculate the variance of new born chickens. From intro stats you may recall that the equation for variance is

$$\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$$

```
> attach(chickwts)
> weight
 [1] 179 160 136 227 217 168 108 124 143 140 309 229 181 141 260
[16] 203 148 169 213 257 244 271 243 230 248 327 329 250 193 271
[31] 316 267 199 171 158 248 423 340 392 339 341 226 320 295 334
[46] 322 297 318 325 257 303 315 380 153 263 242 206 344 258 368
[61] 390 379 260 404 318 352 359 216 222 283 332
```

(This allows us to use the object *weight*, which is the variable of interest)

The long way to calculate variance:

```
> sum((weight - mean(weight))^2) / (length(weight) - 1)
[1] 6095.503
```

And the short:

```
> var(weight)
[1] 6095.503
```

Functions always have at least one *argument* that is stated inside the parenthesis. For functions that have multiple arguments, you can save some time and code if you know the position of each argument. If you know the position, you do not need to specify which argument you are addressing, you just have to put the values in the correct order, you can refer to the seq() and rep() example in the vectors section. The help documentation will always state the order of the arguments.

Other common functions:

plot()	For scatterplots
hist()	For histograms
barplot()	For barcharts/plots
range()	Highest value – lowest value
stem()	For stem and leaf plots
table()	To create tables of categorical variables
pie()	For pie chart

Exercise 4: Write code to standardize and transform the chick weights from the previous example in the following way.

$$log\left[\left(\frac{x_i - \bar{x}}{\sigma}\right)^2\right]$$

where σ is the standard deviation of the weights.