Basic Data Structures in R

Data types and modes

- **`vectors`** (an R data structure consisting of contiguous cells containing data) are the basic building blocks.
- R has six basic (‘atomic’) vector ***types***: logical, integer, real, complex, string (or character) and raw.
  - The R function **`typeof()`** returns the ***type of an R object.***
  - The R function **`mode()`** returns the ***mode*** which is more compatible with other implementations of the S language.

typeof	description	mode
logical	Vector containing logical values	logical
integer	Vector containing integer values	numeric
double	Vector containing real values	numeric
complex	Vector containing complex values	complex
character	Vector containing character values	character
raw	Vector containing bytes	raw

vectors

- The fundamental data structure in R is the **vector**.
- Note: so-called ***scalars***, or individual numbers, do not really exist in R. They are one element **vectors**.
- A **vector** can contain either numbers, strings, or logical values, but not a mixture of data **types**.
- You can create a **vector** from simple elements using the **c(...)** operator:

```
> c(1,2,6,4,8)
 [1] 1 2 6 4 8
> x <- c(88,5,15,44)
> x
 [1] 88  5 15 44
```

vectors

- **vectors** elements are addressable (indexed) by their ***subscripts*** (location) or their names (if they have one):

```
> x <- c(88,5,15,44)
> x[1:3]
[1] 88  5 15
> x <- c(x[1:3],168,x[4])
> x
[1]  88   5  15 168  44
```

- **vector** elements can have ***names***:

```
> v <- c(10,20,30)
> names(v) <- c("Moe","Larry","Curly")
> print(v) # note: same as typing "v" at prompt
  Moe Larry Curly
   10    20    30
```

matrices

- An R **`matrix`** is a two-dimensional numeric array.
- A **`matrix`** is simply a **`vector`** that has dimensions.

```
> A <- 1:6 # A is a vector at this point
> print(A) # same as expressing, or evaluating, A
[1] 1 2 3 4 5 6
> dim(A) <- c(2,3) # Force vector A to be 2 x 3
> print(A) # Now A is a matrix, no longer a vector
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

- Properties of **`matrices`** can be generalized to *n*-dimensional data structures in R as **`arrays`**.

lists

- An R **list** is an ordered collection of objects.
- Unlike **vectors** and **matrices**, **lists** can be heterogeneous (can store objects of different data modes)
- Like **vectors** and **matrices**, you can refer to elements in a **list** by position (by *index* or *subscript*) or by name:

```
> e <- list(thing="hat",size="8.25")
> e
$thing
[1] "hat"
$size
[1] "8.25"
```

e is a **list** with two named components: “thing” and “size”

lists

- Can reference **`list`** components and elements with subscripts:

```
> e <- list(thing="hat",size="8.25")
> e[1]
$thing
[1] "hat"
> e[[1]]
[1] "hat"
> e[2]
$size
[1] "8.25"
> e[[2]]
[1] "8.25"
> e[3]
$<NA>
NULL
```

Single subscript [1] references the first component, named “thing”

Double subscript [[1]] references the elements of the first component

Single subscript [2] references the second component, named “size”

Double subscript [[2]] references the elements of second component

There is no third component to reference with subscript [3]

lists

- Can combine data structures using **list()** function:

```
> x1 <- c(1, 2, 3)
> x2 <- c("a", "b", "c", "d")
> x3 <- 3
> x4 <- matrix(nrow = 2, ncol = 2)
> x4[,1] <- c(1, 2)
> x4[,2] <- c(3, 4)
> Y <- list(x1 = x1, x2 = x2, x3 = x3, x4 = x4)
> Y  # What appears when we type Y at R prompt?
```

What does this **list** structure look like? **x1** is a numeric **vector** w/ 3 elements; **x2** is character **vector** w/ 4 elements; **x3** a numeric **vector** w/ 1 element; and **x4** is a 2 x 2 **matrix**.-

All information contained in the list components of Y is accessible by typing, for example, **Y$x1**, **Y$x2**, and so on. Note that nearly all functions (linear regression, glm, t-test, etc.) in R produce output that is stored as a list.

```
> Y

$x1
[1] 1 2 3

$x2
[1] "a" "b" "c" "d"

$x3
[1] 3

$x4
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

x1 is the first component of **list Y;**
x1 is a numeric **vector** w/ 3 elements

x2 is the second component of **list Y;**
x2 is a character **vector** w/ 4 elements

x3 is the third component of **list Y;**
x3 is a numeric **vector** w/ 1 element

x4 is the fourth component of **list Y;**
x4 is a 2 x 2 (numeric) **matrix**

data frame

- A **data frame** is a **list** that contains multiple named **vectors** of the same length, but which can be different modes.
- Let's construct a **data frame** with the win/loss results in the National League (NL) East in 2008:

```
> teams <- c("PHI","NYM","FLA","ATL","WSN")
> w <- c(92, 89, 94, 72, 59)
> l <- c(70, 73, 77, 90, 102)
> nleast <- data.frame(teams,w,l)
> nleast
  teams  w   l
1   PHI 92  70
2   NYM 89  73
3   FLA 94  77
4   ATL 72  90
5   WSN 59 102
```

data frame

- You can refer to the ***components*** of a data frame (or items in a **`list`**) by name using the **`$ operator`** (or, alternatively, subscripts):

```
> nleast$w
[1] 92 89 94 72 59
```

- Let’s say you wanted to find the number of losses by the Florida Marlins (FLA). You can select any member by using a **`vector`** of Boolean values to specify which item to return:

```
> nleast$teams == "FLA"
[1] FALSE FALSE  TRUE FALSE FALSE
```

- Then you can use this **`vector`** to refer to the right element in the losses column:

```
> nleast$l[nleast$teams=="FLA"]
[1] 77
```