Tutorial 1: R Basics

CS 498 CL1: Probability in CS

Updated September 2nd, 2015

1 First Things First

This tutorial is to familiarize you with R. It goes over basic functions, operator overloading and selecting and reading data. Before you start, make sure you have correctly installed R/R studio. As you go through this tutorial, test out snippets of code yourself so that you can better understand how things work in R.

Also, feel free to use the `help(function)` or `?function` commands at any point to learn more about what a function does. Needless to say, google is your friend.

2 Basics

- *The* `c` *command*

  The `c` command is the easiest way to input a bunch of values to a vector. In order to assign a variable name, use "`<-`". For instance:

```
> wow <- c(5, 234, 23, 566, 56)
> wow
[1] 5 234 23 566 56
> name <- "Doge"
> name
[1] "Doge"
```

- *Support: The* `help` *and* `example` *Functions*

  Call the help function anytime you need more info about certain function arguments, objects etc. In R, this usually opens a browser window, and in R studio, information is displayed within the application. The syntax is `help(function)` and `example("function")`.

```
> help(hist)
# This should open a browser window.
> example("hist")
hist> op <- par(mfrow = c(2, 2))

hist> hist(islands)
Hit <Return> to see next plot:
# Check out what the function does by hitting <Return>
```

You can try out examples for other functions such as `boxplot`, `read.csv`, etc.

- `n:m`

  Create a vector containing values within a range:

```
> amaze <- 24:30
> amaze
[1] 24 25 26 27 28 29 30
```

- Other useful basic commands worth trying out.

```
# These help you maneuver to the R directory you are working in.
# This is important in case you want to load in local csv files.
# In R studio, you could set your directory just by clicking a few
# buttons; your workflow might be different depending on your OS.

getwd() # Gives you the current working directory
setwd() # Change working directory
        # Useful when you need to set your directory where your csv files
        # are located. Again, R studio has a click interface for this.
dim() # Returns the dimension of the variable
```

3 Data Structures

Data structures are usually of the following types:

- *Vectors: Sequence of data elements of a single type.*

```
> my.strings <- c("Hello", "world")              # Vector of length two
> my.strings
[1] "hello" "world"

# Now want to play with vectors of numbers
> a <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)          # Equal to ‘a <- 1:10‘
> b <- c(11, 12, 13, 14, 15, 16, 17, 18, 19, 20) # Equal to ‘b <- 11:20‘
> a+b # Addition, multiplication etc. if vectors of same dimensions
[1] 12 14 16 18 20 22 24 26 28 30

> a*b # Multiplies individual elements, NOT a dot product
[1]  11  24  39  56  75  96 119 144 171 200

# Scalar addition; Logical operators compare each element of a vector
> a + 10 == b
[1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
# NOTE: A vector (or data frame) of booleans is called a mask.
# We won't use them profusely this semester, but they are good to
# know about if you wish to continue learning R.

# To index elements within the vector
> b[4]
[1] 14
# Note that R indexes from 1, rather than 0.

# You can alternatively access a vector with a mask
> mask <- c(T, F, F, F, F, F, F, F, F, T)  # T and F are short
                                           # for TRUE and FALSE
> masked.out <- a[mask]
> masked.out
[1] 1 10
```

- *Data frames: List of vectors of equal lengths; used for storing data tables.*

```
# Create a Data frame for students, identified by
# registration numbers (regnum); with indicated
# marks and a grade for conduct

> regnum <- 1:10
> marks <-c(12, 45, 67, 54, 34, 68, 88, 33, 22, 25)  # This is of length 10
```

```
> conduct <-c("A", "B", "A", "A", "C", "B", "B", "A", "B", "A")

> df <- data.frame(regnum, marks, conduct)
   regnum marks conduct
1       1    12       A
2       2    45       B
3       3    67       A
4       4    54       A
5       5    34       C
6       6    68       B
7       7    88       B
8       8    33       A
9       9    22       B
10     10    25       A

# There are a couple of ways you could access the columns in this
# data frame. We show one here, to not overwhelm you guys.

# The "$" sign notation in R (which you might see now and then) means
# we're accessing an attribute of an object. In a data frame, each
# column counts as an attribute.
> df$conduct
 [1] A B A A C B B A B A
Levels: A B C  # If a non-numeric variable takes on discrete values,
               # these discrete values are called 'Levels'

# Extract data from the data frame using subset()

> subset(df, regnum == 5)
5      5    34       C

# Extract data corresponding to students with conduct grade "A"
> subset(df, conduct == "A")
   regnum marks conduct
1       1    12       A
3       3    67       A
4       4    54       A
8       8    33       A
10     10    25       A
```

4 Reading in Data

You can read in data with commands such as `read.csv()` (Comma delimited files), `read.table` (Usually works for text files and tables)

```
# First, set the correct working directory
> getwd() # get the current working directory
> setwd("<your working folder path here>")

> mydata<- read.csv("filename.csv", header=TRUE, sep=",",)

> head(mydata)
> tail(mydata)
# use these to check whether your data has been imported correctly
```

5 More Resources to Get Started

- R tutor - http://www.r-tutor.com/taxonomy/term/220/0
- R for Beginners - Emmanual Paradis:
  https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf
- Codeschool - http://tryr.codeschool.com/