Module 1: R basics

Jake Ferguson (jakeferg@hawaii.edu)

Module goals

1. Navigate rstudio cloud
2. Understand how data is stored and represented in R
3. Read in datasets from files
4. Run some basic mathematical and statistical operations on the data
5. View and summarize the datasets

R, RStudio, and RStudio Cloud

- "R is a free software environment for statistical computing and graphics". This is the software that actually does the work.
    - download here: https://cran.r-project.org/
- "RStudio is an integrated development environment (IDE) for R, a programming language for statistical computing and graphics." This software is for interacting with R.
    - download here: https://rstudio.com/. You must install R first.
- "RStudio Cloud is a hosted version of RStudio in the cloud that makes it easy for professionals, hobbyists, trainers, teachers and students to do, share, teach and learn data science using R."
    - No download necessary: https://rstudio.cloud/

We will use RStudio Cloud exclusively in this course, but all work can reproduced in R and RStudio.

Navigating `rstudio.cloud`

Assigning values in R

The arrow <- is an assignment operation.

The right-hand side is the value you assign

The left-hand side is the variable you assign a value to.

```
x ← 3.14 #x is a number
y ← "CWS" #y is a character string
```

Anytime we refer to `x` and `y` in the future these values will be available.

```
x
```

```
## [1] 3.14
```

```
y
```

```
## [1] "CWS"
```

Math operations

```
x + 3
```

```
## [1] 6.14
```

```
x*10
```

```
## [1] 31.4
```

```
x/2
```

```
## [1] 1.57
```

but not on characters

```
y*2
```

```
## Error in y * 2: non-numeric argument to binary operator
```

Note that errors in R can be difficult to interpret. Cut and paste into **google** to decipher!

Order of operations

Precedence of operations, from highest to lowest: ^ exponentiation

- / multiplication, division
- - addition, subtraction

However, best practice is to use parenthesis for clarity

```
x ← 36
y ← 7

sqrt(x)*7-2
```

```
## [1] 40
```

```
(sqrt(x)*7)-2
```

```
## [1] 40
```

```
sqrt(x)*(7-2)
```

```
## [1] 30
```

Logical operations

`>` `<` `⩽` `==` `≠` comparison operators (less than , greater thean, less than or equal to, equal to, not equal to)

```
7 < 2
```

```
## [1] FALSE
```

```
7 > 2
```

```
## [1] TRUE
```

```
7 == 2
```

```
## [1] FALSE
```

```
2 == 2
```

```
## [1] TRUE
```

Collections of values

A vector is a collection of values, `c()` combines values into a vector.

```
w ← numeric(10) #A vector of 0's
x ← c(1, 2, 3, 4, 5, 6)  #this is a numeric vector
y ← c('yes', 'no', 'yes', 'no', 'yes', 'no') #this is a vector of characters
```

we can do the math operations on numeric vectors

```
w*1
```

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

```
x*3
```

```
## [1]  3  6  9 12 15 18
```

```
x*x
```

```
## [1]  1  4  9 16 25 36
```

these operations go elementwise, recycling values where appropriate.

Functions

A function takes an argument does some stuff to it, then provides some output.

```
length(x) #returns the length of  vector
```

```
## [1] 6
```

```
print(x) #function to print an object
```

```
## [1] 1 2 3 4 5 6
```

```
mean(x) #calculate the mean of a vector
```

```
## [1] 3.5
```

```
log(x) #calcuate the natural log of a vector
```

```
## [1] 0.0000000 0.6931472 1.0986123 1.3862944 1.6094379 1.7917595
```

Functions for creating vectors

R has some commands to create common vectors quickly that we will use.

```
1:6 #create a sequence of integers with :
```

```
## [1] 1 2 3 4 5 6
```

```
seq(from=1, to=6, by=0.5) #more complex sequences
```

```
##  [1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0 5.5 6.0
```

```
rep(3.14, times=10) #repeat a value
```

```
##  [1] 3.14 3.14 3.14 3.14 3.14 3.14 3.14 3.14 3.14 3.14
```

Excercise 1A

Variables and vectors

Data frames

We can collect vectors into a data frame. These vectors can be of different types.

```
choice.dat ← data.frame(Number=x, Choice=y)
choice.dat
```

```
##   Number Choice
## 1      1    yes
## 2      2     no
## 3      3    yes
## 4      4     no
## 5      5    yes
## 6      6     no
```

Extracting and subsetting dataframes

```
choice.dat[3:5,] #3 rows
```

```
##   Number Choice
## 3      3    yes
## 4      4     no
## 5      5    yes
```

```
choice.dat[,2] #1 column
```

```
## [1] yes no  yes no  yes no
## Levels: no yes
```

```
choice.dat[3:5,1:2] #3 rows and 2 columr
```

```
##   Number Choice
## 3      3    yes
## 4      4     no
## 5      5    yes
```

```
choice.dat$Choice # access columns by na
```

```
## [1] yes no  yes no  yes no
## Levels: no yes
```

Reading in data from a file

Data files are often in the `csv` format (comma-seperated values). We'll learn later about excel files

```
bass.dat ← read.csv(file="BassGrowthOto.csv")
```

The argument file is a string that points to the file you want to load. The above command assumed that the file was in the current working directory (where R loads up). This will always be the case on RStudio Cloud.

If the data file is in a different location, as it often is if you are working on RStudio on your computer, you tell R the filepath:

```
bass.dat ← read.csv(file="C:\\Users\Jake\BassGrowthOto.csv") #A windows path
bass.dat ← read.csv(file="/Users/Jake/BassGrowthOto.csv") #A mac or linux path
```

Looking at your dataframe

Print the first 6 lines of the dataset (printing a large dataframe can freeze your machine)

```
head(bass.dat)
```

```
##   gear yearcap FishID YearClass Age Increment Year
## 1    E    1988      1      1987   1   1.90606 1987
## 2    E    1988      2      1987   1   1.87707 1987
## 3    E    1988      3      1987   1   1.09227 1987
## 4    E    1988      4      1987   1   1.31848 1987
## 5    E    1988      5      1987   1   1.59283 1987
## 6    E    1988      6      1987   1   1.91602 1987
```

Structure of the dataframe

```
str(bass.dat)
```

```
## 'data.frame':    1710 obs. of  7 variables:
##  $ gear     : Factor w/ 2 levels "E","T": 1 1 1 1 1 1 1 1 1 1 ...
##  $ yearcap  : int  1988 1988 1988 1988 1988 1988 1989 1990 1990 1990 ...
##  $ FishID   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ YearClass: int  1987 1987 1987 1987 1987 1987 1988 1989 1989 1989 ...
##  $ Age      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ Increment: num  1.91 1.88 1.09 1.32 1.59 ...
```

Excercise 1B

Reading in data

Special values

Missing values: `NA`

infinite values: `Inf`, `-Inf`

```
1/0
```

```
## [1] Inf
```

logical: `TRUE` and `FALSE`

```
is.numeric(7)
```

```
## [1] TRUE
```

```
is.character(7)
```

```
## [1] FALSE
```

Summary

- R stores numeric and categorical data differently
- `data.frame`s store multiple vectors of data and can contain both numeric and categorical types
- `read.csv` is used to read in csv file, the most common types of input files. These can be exported from excel or libreoffice.