R Workshop week 1: Some Basics

Teal Potter

9/9/2021

Table of Contents

COMMON USES OF R IN OUR SCIENCE FIELDS

Note: the following 4 examples are just examples of what R can do, it's fine if the code makes no sense yet.

R is a calculator

```
1+1  # see end of document for which keyboard keys to click to run a line of code
```

```
## [1] 2
```

```
a <- c(4,6,8)
mean(a)
```

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

Note: just like any calculator, if you make a type you will get a wrong answer or no answer at all

Run statistical tests

```
library(car) #load package that contains Soils dataset
```

```
## Loading required package: carData
```

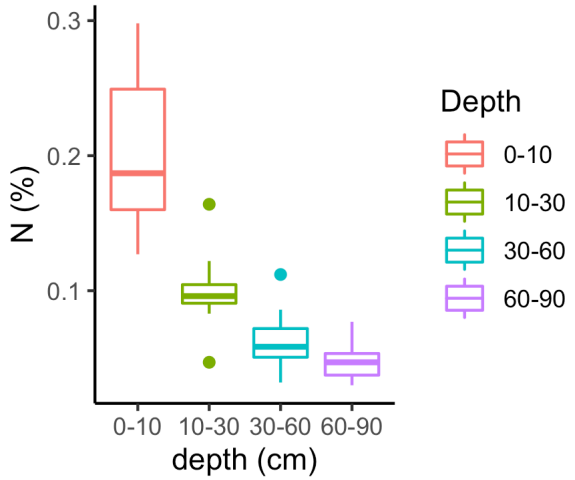
```
library(ggplot2) #load graphing package that contains ggplot() function

summary(lm(pH ~ N, data = Soils))  # run linear model on Soils dataset
```

```
##
## Call:
## lm(formula = pH ~ N, data = Soils)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.63055 -0.42702 -0.01644  0.23039  2.12124
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4.0201     0.1384  29.045  < 2e-16 ***
## N             6.3691     1.1375   5.599 1.15e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5237 on 46 degrees of freedom
## Multiple R-squared:  0.4053, Adjusted R-squared:  0.3924
## F-statistic: 31.35 on 1 and 46 DF,  p-value: 1.149e-06
```

Make figures

```
ggplot(Soils, aes(x = Depth, y = N, col = Depth))+
  geom_boxplot()+
  theme_classic()+
  ylab("N (%)")+
  xlab("depth (cm)")
```



Clean and format data

```
head(Soils) # shows top few rows of dataset
```

```
##   Group Contour Depth Gp Block   pH     N Dens   P    Ca   Mg    K   Na Conduc
## 1     1     Top  0-10 T0     1 5.40 0.188 0.92 215 16.35 7.65 0.72 1.14   1.09
## 2     1     Top  0-10 T0     2 5.65 0.165 1.04 208 12.25 5.15 0.71 0.94   1.35
## 3     1     Top  0-10 T0     3 5.14 0.260 0.95 300 13.02 5.68 0.68 0.60   1.41
## 4     1     Top  0-10 T0     4 5.14 0.169 1.10 248 11.92 7.88 1.09 1.01   1.64
## 5     2     Top 10-30 T1     1 5.14 0.164 1.12 174 14.17 8.12 0.70 2.17   1.85
## 6     2     Top 10-30 T1     2 5.10 0.094 1.22 129  8.55 6.92 0.81 2.67   3.18
```

```
Soils[Soils$Group == 2 & Soils$pH <= 6, 1:6]  # subsetting dataset to be for rows
where group is 2 and pH is less than or equal to 6
```

```
##   Group Contour Depth Gp Block   pH
## 5     2     Top 10-30 T1     1 5.14
## 6     2     Top 10-30 T1     2 5.10
## 7     2     Top 10-30 T1     3 4.70
## 8     2     Top 10-30 T1     4 4.46
```

OVERVIEW OF CODING IN R

Assigning info to objects

An **object** is any unit of data that you assign a name to using <-. In the examples below, x and vect are objects.

```
x <- 2  # assigning the value 2 to be x

x       # run x by itself to check that it is correctly set equal to 2

## [1] 2

x + 4   # now you can use your new object, x, in future calculations

## [1] 6

vect <- c(4,2,0)

vect

## [1] 4 2 0

x + vect

## [1] 6 4 2
```

Using functions to make calculations

For example, + does the same thing as the function `sum()`

Functions are objects that complete and action on data within the following parenthesis. Some functions, like `sum()` can be used any time R is open. Anyone can write their own functions and publish them for people to use (i.e. R is open source) as well. To access functions that are not pre-loaded in R (sometimes called base R) you will have to install small units of software called **packages** to use these functions.

```
1+1

## [1] 2

sum(c(1,1))

## [1] 2
```

Error in sum(vect) : invalid 'type' (character) of argument

It is good to know about modes and classes b/c you will have to troubleshoot why your data didn't behave as you expected

An object can be any type of data

Mode: is a classification of data. Some modes are numeric, complex, character, and logical. R will attempt to recognize numbers (e.g. 2) as mode `numeric` and text (e.g. yes) as mode `character` as default. You can also change the mode of an object

Class: basic structure of units of data.

```
mode(x)

## [1] "numeric"

mode(vect)

## [1] "numeric"

vect <- as.character(vect) # if you name your new object with the same name as
before, you overwrite the first version of vect; it no longer exists. Note
as.character() is a function.


mode(x)

## [1] "numeric"

sum(vect) # because vect no longer contains numeric data, you cannot use a function
like sum that only works on numeric data. You will get an error message.

## Error in sum(vect): invalid 'type' (character) of argument
```

Basic classes of data

- **vector**: a single column of data (1 x length n)

```
vect <- c(4,2,0)

vect

## [1] 4 2 0
```

- **matrix**: a dataset that only contains numerical data

```
mat <- cbind(vect, rep = 3)

mat

##      vect rep
## [1,]    4   3
## [2,]    2   3
## [3,]    0   3
```

```
class(mat)

## [1] "matrix" "array"

mode(mat)

## [1] "numeric"
```

- **data frame**: a dataset that can contain many modes of data

```
char_vect <- c("yes", "no", "yes")  #making new vector that contains text instead
of numbers

df <- data.frame(mat, char_vect) # adding this new text vector to my matrix turns
it into a data frame

df

##   vect rep char_vect
## 1    4   3       yes
## 2    2   3        no
## 3    0   3       yes

class(df) # check what class df is now

## [1] "data.frame"

# to check mode of each column in the dataframe, select the arrow next to the
object under teh environment tab (most likely in the top right quadrant of Rstudio)
```

SOME BEST BEST PRACTICES WHEN GETTING STARTED

Work in Rstudio

There are lots of useful features to keep track of what you are doing

Save your script often

The key to making progressing is baby steps. Save your file every time you get a piece of code to work when you are starting out.

Access resources within RStudio

Type in a function name or pre-loaded dataset's name into the Help tab in the lower right quadrant of RStudio on your screen or simply run a line of code that is just the name with a ? before it (no spaces). This reveals help documentation in the lower quadrant of RStudio

```
?Soils
```

Set preferences for Rstudio layout

There are many ways to personalize your R studio experience. Too see a full menu of preferences go to the R Studio tab on the bar at the very top of your desktop (title bar). Navigate to "preferences" in the dropbox menu. Here, you'll find settings including fun options like color/font preferences (under "appearance") and practical ones like soft-wrap (under "code", makes it so that your code will automatically start a new line to fit in the script window).

Keyboard actions

Description	Windows & Linux	Mac
Run current line/selection	Ctrl+Enter	Command+Enter
Comment/uncomment lines of code	Ctrl+Shift+C	Command+Shift+C

Shortcuts resource: https://libguides.libraries.claremont.edu/c.php?g=480755&p=3350989

More beginner info on loading data, saving objects produced in R, and basic functions

https://tealpotter.weebly.com/teaching.html Open 'INTRO TO R' pdf