Introduction to R
Philip Dixon, 13 August 2022

Access to R and Rstudio
R is open-source, free, community-maintained software. It is installed on all Statistics Dept. terminal servers, so it is available in Snedecor 1105 and 3121 and Gilman 2272. It has been installed in many departmental and ISU computer rooms.

RStudio is an integrated development environment developed by the RStudio corporation. It is (for now) free to individual users.

Both R and RStudio are available for Window, Mac and Linux (3 flavors) operating systems. Lab will be much more useful if you have R and RStudio installed on your laptop.

Install R first, then install RStudio.

To install R on your laptop, download it from a CRAN (Comprehensive R Archive Network) site. The closest mirror site is `mirror.las.iastate.edu/CRAN/`. Look for the `Download R for XX` link for your operating system in the "Download and Install R" window at the top of the page.

To install RStudio, visit `www.RStudio.com` and click either the Products drop-down menu and choose RStudio, then RStudio Desktop. Or, click the download link (not very obvious) and choose RStudio Desktop.

These instructions assume you will be using RStudio. If you don't like that IDE, use R instead. But, there are lots of useful features in RStudio that are not available in "vanilla" R. I usually enter commands into RStudio so the code I provide and describe will tell you the command to do something in R even though there is a menu item or button in RStudio to do the same thing. I illustrate this in the section on **Reading data into a data frame**.

Using RStudio and R: RStudio layout
Start RStudio by double-clicking the RStudio icon. You will see three (later four) windows:
Console: This is where you type R commands and where R provides output.
Environment/History:
The Environment side tells you the names of all the objects (data frames, saved output, etc.) in your workspace. If you click on the name of a data frame, a fourth window opens with the contents of that data frame.
The History side gives you a scrollable window with all the commands you've entered. This is saved from session to session, so you can see or repeat what you did last week/month/year.
Files/Plots/Packages/Help/Viewer:
Files: a list of all files in your working directory
Plots: the current plot
Packages: a list of add-on packages (libraries) available to you and (as User buttons) activated.
Help: When you request help on a function, the help file appears here.

When you open a file of R commands, the contents of that file appear in a fourth window, the script window.

Using RStudio and R: Interacting with R
The main way of interacting with R is to type a command into the Console window (> prompt). The result of that command are returned in the Console window. Or, you can execute commands from the script window. The third approach is to run an entire file of commands ("sourcing a file"). We will focus on the first two approaches.

Exercises - 1

- Click the mouse in the Console window, then type 1+2 and hit the enter key. R should return the result: 3
- Hold down the Shift and Ctrl keys and type n, or select File / New File / R Script from the main menu. Either opens a Script window (top left). Click the mouse in that window to put the cursor there, if it isn't already. Type 1+2 and hit the enter key. Nothing should happen, except for a new line number. Use the mouse or up arrow to put the cursor on line 1 (1+2), then hold the Ctrl key and hit the enter key. This executes the selected line by copying it to the console window and running the command.

Getting files from the class web page
I will distribute files of R commands and data sets on the class web site. To get R command files, you need to download the file from the web site, make note of where that file was stored, then load it into the Script window. To load data sets, you can either download the file from the web site and read it into R, or read it directly from the web site. Details of the default download directory depend on your operating system, browser, and browser settings. All I can say is you will need to know the directory / folder where the file(s) you download are stored.

R will look for files in a default folder. That location depends on the Windows / Mac / Unix choice and may depend on the specific version. You can determine that default folder by typing `getwd()` into the Console window and hitting enter. You can change that folder using the `setwd(choose.dir())` command or selecting Session / Set Working Directory from the main menu. `choose.dir()` opens a window to browse to a folder (**dir**ectory). You can also type the relative or full path to the desired default folder, e.g., `setwd('stat587')`, `setwd('../Desktop')`, or `setwd('c:/users/pdixon/Documents/stat587')`. Note: these are names for parts of the Windows 10 file structure. Mac and Linux systems have different names for parts of their file structures. So may Windows 11.

Heads-up: In previous years, folks have had a lot of trouble figuring out how to get your browser to store files where R can find them with a name that you can use. This is a browser/operating system problem. There are lots of ways you can get this to work. Do whatever works for you. Have patience; once figured out, you're good for the semester (and probably forever).

Data sets in R A variable in R may contain a constant, a vector, a matrix, a function, or a data frame. Data frame is the R name for a data set that has been stored in R. The data set is organized with a column for each variable and a row for each observation. It is common (and good data management practice) for the names of the variables to be in the first row. This row is removed when the data set is stored as a data frame.

Before you can use a data set for any computations or graphs, you have to read the data set into a data frame. The appropriate commands depend on the file type (e.g., comma delimited, space delimited, excel worksheet). For a CSV file, you can:
1) use the read.csv() function after downloading a file
2) use File / Import Dataset / From CSV from the main menu.
  This also allows you to provide a URL to read directly from the class web site, if saving the file is too troublesome.

For a text file with spaces between fields, you can:
1) use the read.table() function after downloading a file
  If the first line has variable names, add header=T to indicate this.
2) use File / Import Dataset / From CSV from the main menu.
  In the wizard box, change the column delimiter to whitespace.
Code to read text files will be demonstrated next week.

The first line of creativity1.r illustrates 1) for a .csv file. The `<-` assigns the quantity on the right-hand side to the variable named on the left. The required argument to `read.csv()` is the name of the file to read. The other argument is useful. `as.is=T` says to leave character strings as they are and not convert them to factors. Discussion of factors and why you might not want to automatically convert to a factor is postponed to later.

You can use any name you want for the variable (`creativity`, on the left side of the assignment). It does not have to match the name of the data file. However, the file name has to match the file name in your working directory. The two additional arguments have to be spelled exactly as given, although sometimes R will accept the first few letters when they are unique. The `T` is short-hand for "TRUE".

If you use 2), you see a window that allows you to name or browse for a file or a URL. When you enter or choose a file and click update, R brings up a window showing how the data will be read by R. If the file wasn't read correctly, you can change options until it is read correctly. You may want to change the name of the data frame (Name in Import options) if you don't like the default name. When you click "Import", RStudio will create the appropriate commands in the Console window and execute it. The subsequent View(name) in the Console window opens the named data frame in a window (top left).

Exercises - 2

- Download the creativity.csv file from the class web site and make sure it is in your R working directory.

- Enter `read.csv('creativity.csv', as.is=T)` into the console window. If you get a file not found error, the creativity.csv file is not in the R working directory, or you mistyped the name. You should see a lot of rows of data in the console window. You just read the data file but didn't store it, so R prints it by default.

- Enter `creativity <- read.csv('creativity.csv', as.is=T)` into the console window. You should just get a new command prompt (>) in the console window. You may notice that `creativity` was added to the list of Data Frames (top right window).
- Enter `view(creativity)` into the Console window. The contents of the creativity data set should appear in the top left window.
- If you want to try out more things, read the data file using the File / Import Dataset menu.

Long lines
Often, you need to enter a command that is longer than one line. This happens when you have to specify a lot of options. Also, it is often easier to read a command when each argument is on a separate line.

R assumes everything you type is part of one command until everything is "completed". That is all character strings have ending quotes and all function calls have closing parentheses. If you see + after typing a command, that means R is expecting more. Something isn't completed yet. Figure out what is missing and provide it.

RStudio provides auto-completion by default. That means when you type a quote or parenthesis, the matching end quote or parenthesis is provided automatically. What you type after typing the quote or parenthesis is inserted in between the quotes or parentheses.

You can also enter multiple commands on one line. Just end each command with a semicolon (`;`).

Referring to variables in a data frame You will often need to identify specific variables in a data frame. The creativity data set has two variables (the treatment and the score, for each individual). Because the first line of creativity.txt has the words treatment score, the first variable is named treatment (first column of data) and the second variable is called score (second column of data). R looks at the first few lines of the file to decide whether a variable is numeric (numbers) or character (words).

If you followed the directions above, the data set is stored as a data frame called creativity. To access a variable in a data frame:

- name the data frame and variable all at once with a `$` between the two pieces. `creativity$treatment` is the treatment variable in the creativity data frame. This approach always works but it can be cumbersome, especially when referring to more than one variable in the same data frame.

Some R functions accept two other ways to reference a variable in a data frame:

- name the variable and where to find it. `treatment, data=creativity` is the treatment variable in the creativity data frame. This approach only works with functions that support the `data=` argument. This approach is especially useful for functions that support the formula interface.

- name variables in a formula. We will frequently want to model a response variable as a function of some predictor or grouping variables. This is conveniently specified as a formula. For example, to look at scores (response) in each treatment (grouping variable), you could specify `score ~ treatment, data=creativity` to a function that accepts a formula. The response is on the left side of the `~`; the grouping/predictor variables are on the right side. You could specify the variables without using data=, by `creativity$score ~ creativity$treatment`, but all functions that accept formulae are supposed to accept data=, which simplifies and clarifies what you write.

Examples of all of these ways of referring to variables are in `creativity.r`.

Capitalization This **really really matters** in R. `treatment` is a different variable than `Treatment` which is different from `TREATMENT`. This applies to all names in R: function names, data frame names, variable names, strings. I will usually use all lowercase variable names. Some datasets come with capitalized names of various sorts. If something doesn't work, check the capitalization.

Quitting R There are three ways to quit R:

- Enter `q()` into the console window. This runs the quit function, q.
- use File / Quit R Studio from the menu bar.
- hold down the Ctrl key and type q. This is the hot-key for File / Quit.

In all three, you will be asked whether to save `.Rdata`. You should. This saves your workspace. The next time you start R, the workspace is loaded automatically, so you can resume your work where you left off.

Getting help
Each R function has a help file that tells you what the function does, all the arguments, and all the output. You see the help file by entering `?function` or `help(function)`, where `function` is replaced by desired function, in the command window. For example, `?hist` or `help(hist)` displays the help file for the hist() function in the lower right window.

The help files are often cryptic. Even so, if you aren't sure about something, look at the help information. It may answer your question.

There are many introductions to R around the web. The CRAN site, e.g.
`mirror.las.iastate.edu/CRAN/` has "official" documents (see the Manuals link under Documentation in the left frame) and user supplied documents (See the Contributed link). If you look at the official documents, the Introduction to R is likely to be the most useful, but I find it has more details than a beginning user needs. The class web site has links to a more useful tutorial from the University of Georgia. There are many others floating around the web.

R Studio has links to the official documents. To see the introduction to R document, go to Help / R Help / Introduction to R from the main menu.

Notes and Comments:

- The `#` character is the R comment character. Everything from the `#` to the end of the line is ignored. Comments may be on a line by themself or to the right of an R command. Class R files include brief comments. An extended discussion of the examples is in the explanation file for each code file.

- Quotes surround character strings, like file names. R accepts either single (`'`) or double (`"`) quotes. They just have to match, so a character strings that starts with a single quote is ended by a single quote. The backquote (key to left of 1) is not used.

- The `<-` symbol for assigning to a variable dates from the earliest versions of R and its predecessor, S. That was chosen, I am told, so that three different interpretations of = (assignment, logical equality, argument naming) were kept separate. R now accepts = for assignment, so `creativity = read.table('creativity.txt', header=T, as.is=T)` does exactly the same thing as `creativity <- read.table('creativity.txt', header=T, as.is=T)`. You may use either. Class code uses `<-` because that's what Dr. Dixon's fingers learned years ago.

- Plots in R can be drawn using at least three different graphics systems. My code will illustrate base graphics. You are welcome to use ggplot graphics if you are familiar with it. The lattice graphics system is much less frequently used.

- Any specific task can almost always be done more than one way in R. Class code illustrates what has worked for Dr. Dixon. If you prefer to use another way to do something, feel free to, after checking (by a test example, or with Dr. Dixon) that your way is equivalent.