Lab 1: R Basics and Working with Data

Stats 12

Main exercises based on labs by Nicolas Christou and Robert Gould.

Submission Instructions

1. Save your final answers in PDF format.
2. Upload your PDF to the submission link on CCLE.
3. Submit the assignment before the deadline. No late assignments will be accepted.

Objectives

1. Demonstrate basic R Skills (create vectors, perform vector operations, install packages)
2. Use basic statistical functions (mean, min, max, median, sd)
3. Visualize data (dot plot, histogram, box plot, contingency tables, scatter plot, bubble plot)

Before Coming to Lab

1. Download the latest version of R at http://cran.stat.ucla.edu/
2. After R is installed, download the latest version of RStudio (the free one on the left): https://www.rstudio.com/products/rstudio/download3/
3. Download "Introductory Statistics with R" by Peter Dalgaard. The book is available for free on springerlink.com. You must access it via a campus computer or using the BOL VPN.
4. Read 1.1 and 1.2.1-1.2.3 of the Dalgaard book
5. The computer lab is equipped with all the tools necessary to complete your lab assignments. However, if you prefer to work on your own computer you can bring your laptop to the lab as well.

A Primer on labs

In Stats 12 the lab courses are designed to help you explore the capabilities of computer programming to assist you in statistical analysis. This quarter we will be using RStudio, a high-level statistical programming language. We will cover several basic topics in RStudio to help you carry out analysis that you learn in lecture. RStudio is a popular program in universities because it is free and open-source, which means anyone can access the program's source code and suggest improvements.

Collaboration Policy

In Lab, you are encouraged to work in pairs or small groups to discuss the concepts on the assignments. However, DO NOT copy each other's work as this constitutes cheating. The work you submit must be entirely your own. If you have a question in lab, feel free to reach out to other groups or as the TA if you get stuck.

Section 1 – R and RStudio Basics

Instructions

When you open RStudio, you'll notice the window is partitioned into four sections:

- **The top left section** is your working code file. This is where you should type and run your commands. It serves two main purposes:
    - You can edit and save this section just like a word document. In fact, let's save right now. Go to file>save, and save this in a convenient location on your computer.
    - You will run lines of code by highlighting the code chunk you want to run, and clicking the run button (shortcut: ctrl+enter for PC, cmd+enter for Mac)
- **The bottom left section** is the "console". This is essentially the R engine which runs your commands and produces output. You should not need to type anything in this section, but you will need to copy and paste output for exercises.
- **The top right section** defines your environment. The environment contains any objects you create and data you import. You won't use it much but it's a quick way to see what data and objects you have to work with.
- **The bottom left section** contains three important tabs in this course:
    - **Plots:** when you create plots, they will show up here. You should copy and paste these to answer some of the lab exercises.
    - **Packages:** R contains packages that you can install and run to use cool features and functions that the basic R install did not give you.
    - **Help:** If you are unsure about what a function does, try searching for it in the help tab. You will find information about the function arguments and examples.

Working with packages

To work with packages, we must do two things:

1. Install the package by using the packages tab mentioned above
2. Type and run library("package_name") to tell R to load the package we installed

Reading Data into RStudio

R needs to be told which folder to look in to retrieve data. To read in data, first download it into a convenient folder on your machine. Then, try one of these methods:

1. At the top of RStudio, go to Session> set working directory> choose directory. Then select the folder the data is saved in, and click "Select Folder". Then, create a new data object by using the syntax object_name <- read.csv(file = "filename.csv")
2. Use the syntax object_name <- read.csv(file.choose()) and use the file explorer to manually select the downloaded data.

Vectors – Examples

Try running the commands below. The first creates a numeric vector object called *numbers* that contains the numbers 1 to 5. The second creates a character vector object called *schools* that contains the names of two top-tier colleges, and the name of a terrible school for spoiled children. Note that when you create a vector of names, you must put quotes around each element. The c() function is used to create both vectors. Try typing c() in the help menu to get a better sense of what this function does.

numbers <- c(1,2,3,4,5)

schools <- c("UCLA", "UC Berkeley", "USC")

Note that when you run these commands, there is no output. That is because whenever you define an object, there is not output. However, once you have made the object, you can use the print function to print contents of an object. Example: type print(numbers)

If the vector is numeric, we can do math on it. For example, try typing numbers*2

We can also use square brackets to create subsets of our data. For example, if you type schools[2] you will retrieve only the second element from the schools vector.

Object classes

Vectors, matrices, and data frames are three common object classes you will work with. Matrices and data frames can be thought of as tables of data. However, matrices contain only numeric data. Data frames can contain several types of data. The file you will work with in this lab is a sample of information about babies born in North Carolina. It is considered a data frame because it contains numeric information about each baby, as well as various pieces of categorical data such as the race of the parents, and whether the mom has a smoking habit.

Section 1 Exercises

1. - vectors:

   a. Create a vector named *heights* that contains the heights, in inches, of yourself and two students near you. Print the contents of this vector.
   b. Create a vector named *names* that contains the names of these people. Print the contents of this vector.
   c. Try typing cbind(heights, names). What did this command do? What class is this new object?

2. – downloading data:

   a. Download the data set births.csv from the CCLE site and upload it into Rstudio. Name the data frame *NCbirths*.
   b. Demonstrate that you have been successful by typing head(NCbirths) and copying and pasting the output into your word processing document.

3. – Load the maps package

   a. Install the maps package. Verify its installation by typing find.package("maps") and include the output in your answer.
   b. Type library(maps) to load up the package. Type map("state") and include the plot output in your answer.

4. – Perform vector operations

   a. Extract the weight variable as a vector from the dataframe by typing
      weights <- NCbirths$weight
   b. What units do you think the weights are in?
   c. Create a new vector named weights.in.pounds which are the weights of the babies in pounds. You can look up conversion factors on the internet.
   d. Demonstrate your success by typing weights.in.pounds[1:20] and including the output in your word processing document.

Section 2 – Summarizing Data (one variable)

Useful functions

The functions summary(), mean(), sd(), max(), and min() all produce helpful results to help us understand how quantitative data is distributed. These functions will take in a numeric vector inside the parentheses. As an example, max(NCbirths$weight) will give us the maximum weight of all the babies in out sample.

We can also use the tally() function in the Mosaic package to help us summarize categorical data. Tally requires a categorical vector inside the parentheses. For example, after you have installed and loaded the Mosaic package, try typing tally(NCbirths$Racemom).

Section 2 - Exercises

1. What is the mean weight of the babies in pounds?
2. What percentage of the mothers in the sample smoke? *Hint: use the tally function with the format argument. Use the help screen for guidance.*
3. According to the Centers for Disease Control, approximately 21% of adult Americans are smokers. How far off is the percentage you found in question 2 from the CDC's report?

Section 3 – Visualizing Data (one quantitative variable)

Useful functions

The histogram, dot plot, and box plot are useful tools for visualizing quantitative data. The functions you can use in R are histogram(), dotPlot(), and boxplot(). Browse the help screen for these functions to get a sense of what these functions require.

Section 3 - Exercises

1. Produce a dot plot of the weights in pounds.
2. Produce three different histograms of the weights in pounds. Use 3 bins, 20 bins, and 100 bins. Which histogram seems to give the best visualization, and why?
3. We can use the syntax boxplot(vector1, vector2) to make a side by side box plot. Create a side by side boxplot of the mother's ages and the father's ages. Which gender tends to be older?
4. Try typing histogram(~weight | Habit, data = NCbirths, layout = c(1,2)). Describe what this code does. Based on the graph, do you see any major differences between baby weights from smoking moms vs. non-smoking moms?

Section 4 – Visualizing Data (two categorical)

Useful functions

With two categorical variables we can again create two-way tables using the tally() function. For example, try tally(~Habit | MomPriorCond, data = NCbirths, format = "proportion")

Section 4 - Exercises

1. Consider the other categorical variables in this data. Of those that record the health of the baby, which do you think will be associated with the mother's smoking and why? Make a two-way Summary Table to check your hypothesis. Do you have evidence that this variable associated with smoking? Why?

Section 5 – Visualizing Data (two quantitative)

Useful functions

To visualize a potential relationship between two quantitative variables, we typically use scatter plots. The syntax is plot(variable1~variable2)

Also, for most plotting functions we can add some arguments to make the plots nicely formatted and look more presentable.

- The *col* argument changes the color of the data points.
- *cex* changes the size of the points
- *pch* should be an integer from 1 to 25 and changes the style of the points
- *xlab* and *ylab* define the labels for the x and y axes
- *main* gives us the title for the plot

As an example, run and compare the following lines of code:

```
plot(NCbirths$weight ~ NCbirths$Gained)

plot(NCbirths$weight ~ NCbirths$Gained, col = "red", cex = 1.5, pch = 3,
    xlab = "Weight gained during pregnancy", ylab = "Baby weight (oz.)",
    main = "Baby weight vs. pregnancy weight gain")
```

Most of these arguments work for histograms, box plots, and dot plots too!

Section 5 - Exercises

1. Produce a nicely formatted scatter plot of the weight of the baby vs. the mother's age.

Section 6 – Visualizing Data (geographic data)

If we have geographic data, often the best way to visualize it is with a bubble plot. Please read pages 8-11 of the handout titled “stat12_intro_to_R”. Read in the data as instructed, and replicate the code to reproduce the plots.

Remember that the maps package needs to be installed and loaded before beginning this exercise.

Section 6 - Exercises

1. To demonstrate you have followed the handout, produce a modified version of the colored bubble plot. Rename the title to "California ozone bubble plot". Also, use a different point style, and different colors for each air quality category.