

Lab 1: Getting Started with R and EDA

Objectives:

1. Use RStudio to read and examine a data set.
2. Become familiar with R Markdown files and create a PDF from one.

Definitions:

- variable vs. observation
- quantitative vs. categorical variables
- bar chart, contingency table, histogram
- center, shape, spread of a distribution
- sample mean, sample variance, sample standard deviation, sample range
- percentile (aka quantile); first, second, and third quartiles; median
- 5-number summary, boxplot

Introduction:

One of the goals of this course is to help you gain computational fluency with a statistical language. We will use the language R in this class. R is a powerful, widely used statistical language that is free and open source. The purpose of this lab is to help you get started with this language by performing *exploratory data analysis* (EDA). Exploratory data analysis is an approach to examining and describing data to gain insight, discover structure, and detect anomalies and outliers.

Activities:

Getting Organized: The first goal is to get organized. If you haven't already done so, make a folder on your laptop or in your H: drive (personal network drive on Saint Mary's campus, for info see http://sites.saintmarys.edu/~resnet/..._10resnet.html) that will store all your materials for this class. Make a subfolder called "Labs", and within "Labs" make another subfolder called "Lab1". Finally, download the lab notebook and data file for this lab into your "Lab1" folder. (The files are on the class Blackboard site.)

Configuring RStudio: The next step is to configure RStudio. Open RStudio by double clicking the RStudio icon. By default, RStudio has four "panes", and these can be configured. Click on "RStudio > Preferences... > Pane Layout", and see what your configuration is. I like to have the following configuration:

- Upper Left: Source
- Lower left: Console
- Upper Right: Environment, History, etc.
- Lower Right: Files, Plots, etc.

Set your pane configuration to match this, if it doesn't already. Once you have done so, find and click on the "Files" tab in the lower right panel. Navigate to the "Lab1" folder you just created. Then click the "More" button in the file browser (i.e., the menu at the top of the lower right panel in RStudio) and click "Set as working directory".

Open the Lab Notebook: Before you can start experimenting, you need to set up your "lab notebook" that will store your conjectures, responses to reflections, and any results. You will be using R Markdown files to create your lab notebooks. I have provided a template for you to use on this first lab, it is the "lab1_notebook.Rmd" file you have downloaded and saved in your Lab1 folder. To open it, click on it in the file browser in the lower right pane of RStudio. The document will open in the upper-left pane.

You will edit the "lab1_notebook.Rmd" file as you work through the lab. To start, add your name as the author in line 3 of the document. Once you have completed the lab, you will "Knit" the .Rmd file which will render a PDF for you to turn in. You can try this now by clicking the "Knit" button.

Running R By Command Line: Our next task is figure out how to "run" R. The simplest approach is to enter commands one at a time on a command line. If you've configured your panes as above, there should be a "Console" tab in the lower left pane. Click on it. You should see a ">" prompt. This is where you type commands and then hit "Enter" or "Return" to run them. Try the following:

```
1 + 2

## [1] 3
```

The answer "3" should materialize on the screen. In other words, you can use R like a calculator. Try something more advanced:

```
sqrt(3^2 + 4^2)/5

## [1] 1
```

You can save the output of these calculations in variables. For example, try the following:

```
x = sqrt(3^2 + 4^2)/5
```

Note that this time, you don't see any output when you run the above line. But if you type `x` in the Console, you will see that `x=1`. In other words, you have defined a variable in the workspace and assigned it a value. If you look at the upper right pane in RStudio and click on the "Environment" tab, you should see a list of all the variables in your workspace. (At the moment there should only be one, namely `x`.)

Pause for Reflection #1:

Suppose you didn't know that `sqrt()` was the square root function in R, but you needed to take a square root. How might you figure out that this function existed? (Hint: How do you figure *anything* out in this day and age?) Type a sentence in your "lab1_notebook.Rmd" file in RStudio that explains your approach. Then test your theory by trying to find the R function that gives absolute value. Make a note in your lab notebook about the results of this attempt.

Running R From Scripts: If you want to enter a lot of commands, it is easier to type them all in a single file and then tell R to "run the script", i.e., execute the commands in sequence, one at a time. To generate a script in RStudio, click "File > New File > R Script". In the upper left pane a new blank tab entitled "Untitled1" should materialize. The first thing to do is to give this document a proper name. Click "File > Save As", and save it as "<YourFirstName>_lab1.R".


To test out your script, write the following lines in the document:

```
y=3
z=2
```

Then put your cursor on the `y=3` line and click the "Run" button, which is in the right-hand corner of the pane. You should see the command echoed in the Console pane, and the variable `y` will show up in the Environment tab. Now put your cursor on the `z=2` line and repeat - once again, the new variable should be reflected in the Environment tab.

If you want to execute the two commands together, in one fell swoop, you can click the "Source" button. "Sourcing a script" sends the entire script to the Console to be run. You can also select a portion of lines in a script with your mouse and then, with the lines highlighted, click the "Source" button.

Running R in R Markdown Files: What if you would like to include some code in your lab notebook? Well you would be in luck! One of the reasons I am having you use R Markdown files to create your lab notebooks is because they offer the flexibility to combine text (including LaTeX!) and code in one place. To add code to an R Markdown file insert a *code chunk* either by typing the chunk delimiters directly into the R Markdown file or by using one of the following shortcuts:

- keyboard shortcut: **Ctrl + Alt + I** (Mac **Cmd + Option + I**)
- RStudio shortcut: click the "Add Chunk" button  in the editor toolbar of the .Rmd file

When you "Knit" an R Markdown file, the code in any code chunks will be run and the results will be displayed in the resulting document.

Pause for Reflection #2:

The "lab1_notebook.Rmd" file already contains two code chunks in the Reflection #2 section. Each chunk has the same code, but note that the second chunk has been customized with the additional argument in the chunk header, i.e., the `echo=FALSE` set in the `{ }`. Knit the file and inspect the rendered PDF.

Add a third code chunk with the same code as the other two to your lab notebook and add the following argument: `include=FALSE`. Knit the file again and inspect the rendered PDF.

Explain what the effects of the `echo=FALSE` and `include=FALSE` arguments are on the rendered PDF. Type your explanation below the code chunks in the "lab1_notebook.Rmd" file.

Loading Data: To do statistical analysis on a computer, you need to have some way of getting your computer to read and store datasets. In R there are a number of ways to do this. It is often the case that you find data in a spreadsheet, or simply stored in a regular text file. In fact, the data file for this lab is a .csv file. To load it, write the following command into your R script:

```
FlightDelays = read.csv("FlightDelays.csv", header = TRUE, sep = ",")
```

and click the "Run" button. With any luck, you should see a new variable in your Workspace (i.e., Environment tab in upper right pane) called "FlightDelays", along with a tagline that says "4029 obs. of 10 variables". Click on this data in the Environment tab, and in the upper left pane you should be able to view the data.

Pause for Reflection #3:

Note that the name of the data file is the same as the name of the variable used to store the data in RStudio. Suppose that you didn't want to work with a variable called `FlightDelays`, and instead wanted to work with a variable simply called `data`. How can you change the variable name? Write the commands in your lab1.R script, and execute them. (Hint: First copy it, then remove it.)

Create a code chunk in your "lab1_notebook.Rmd" file and add the code to change the name of the dataset to `data`.

Examining Datasets: Some Key Terms Take another look at the dataset (click on it in the Environment tab to get it to display in a new window). It describes information on 4029 departures of United Airlines and American Airlines from LaGuardia Airport during May and June 2009.

Each row of the dataset is an *observation*. Each column represents a *variable* - some feature or characteristic obtained for each observation. To view the names of the variables in a dataset, try the following:

```
names(data)
```

There are two types of variables:

- *quantitative* (aka numerical) - variables that have numerical values and arithmetic operations are meaningful
- *categorical* (aka factor) - variables that have non-numerical values or numerical values but arithmetic operations are **not** meaningful

Pause for Reflection #4:

Consider (with your partner) what these things are in terms of this dataset. Then in your lab notebook, respond to the following questions:

1. How many observations are there?
 2. How many quantitative variables are there?
 3. and how many categorical ones?
-

Tables and Bar Charts: The categorical variable `Carrier` in the dataset assigns each flight to one of the two airlines: `UA` for United and `AA` for American. To obtain a summary of this variable, try the following:

```
table(data$Carrier)
```

To visualize the distribution of the `Carrier` variable, create a *bar chart*:

```
barplot(table(data$Carrier))
```

We can also use the `table()` function to investigate the relationship between two categorical variables. The following creates a *contingency table* to investigate the relationship between the airline (`Carrier`) and whether or not a flight was delayed more than 30 minutes (`Delayed30`):

```
table(data$Carrier, data$Delayed30)
```

Pause for Reflection #5:

In your lab notebook, create a code chunk that will include the tables and bar chart that we just produced in your rendered PDF. Comment on the distribution of the carrier data and its relationship with delays.

Histograms, Numerical Summaries, and Boxplots: Now focus on the quantitative variable `FlightLength` in the dataset, which gives the length of flight time in minutes. Because it is a bit cumbersome to use the syntax `data$FlightLength` each time we want to work with the `FlightLength` variable, we can streamline things by giving it a new name:

```
f1 = data$FlightLength
```

To see the distribution of a numeric variable, we create a *histogram*:

```
hist(f1)
```

Pause for Reflection #6:

In your lab notebook, create a code chunk that will include the histogram that we just produced in your rendered PDF.

Inspect the histogram and comment on the *center*, *shape*, and *spread* of the flight length data.

The histogram is great, but we'd like some hard numbers, too. We start with the definitions of some commonly used *sample statistics*.

Definition 1.1: Let $\{x_1, x_2, \dots, x_n\}$ be n data points, i.e., a set of quantitative data collected from a sample of the population.

1. *sample mean*: $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$
2. *sample variance*: $s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$
3. *sample standard deviation*: $s = \sqrt{s^2}$
4. *sample range*: $\max\{x_1, \dots, x_n\} - \min\{x_1, \dots, x_n\}$

Definition 1.2: Let $0 < p < 1$. The $(100p)$ th percentile of a set of quantitative data is a number, denoted π_p , that is greater than $(100p)\%$ of the data values. In particular, we have:

- *first quartile* (25th percentile): $Q_1 = \pi_{0.25}$
- *second quartile* or *median* (50th percentile): $Q_2 = m = \pi_{0.5}$
- *third quartile* (75th percentile): $Q_3 = \pi_{0.75}$

The *5-number summary* of a quantitative data consists of the minimum value, Q_1 , m , Q_3 , and maximum.

The sample statistics defined above can be computed for the flight length data as follows:

```
mean(f1)
var(f1)           # sample variance
sd(f1)           # sample standard deviation
max(f1)
min(f1)
range(f1)
median(f1)
quantile(f1)      # quartiles
quantile(f1, 0.3) # 30th percentile
summary(f1)      # 5-number summary & mean
```

Boxplots provide a visualization of the 5-number summary. Try the following to generate a boxplot of the flight length data:

```
boxplot(f1, horizontal = FALSE) # change horizontal = TRUE to change orientation
```

Pause for Reflection #7:

In your lab notebook, copy the boxplot and write down the 5-number summary for the flight length data. Compare the two and discuss with your neighbor how boxplots are constructed from the 5-number summary.

Can you tell just from the boxplot whether or not the data is skewed? Comment. Which is bigger, the mean or the median? In what direction is the data skewed? With your neighbor, debate the claim that "if the data is skewed to the right, the mean is pulled to the right of the median." Record your thoughts in your lab notebook.

This page titled [Lab 1: Getting Started with R and EDA](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Kristin Kuter](#).