R Swirl

Antoine Beauchamp

January 6th, 2017

$$E = mc^2$$

Introduction

Login: abeaucha

`...` means press Enter

Exit swirl at any time by pressing Esc.
At the prompt, type `bye()` to exit and save your progress.

Some commands to use while following a course:
`skip()` allows you to skip the current question
`play()` lets you experiment with R on your own
`nxt()` will allow you to regain swirl's attention.
`main()` returns to swirl's main menu
`info()` displays these options.

swirl allows you to learn courses on R directly in RStudio.
Courses can be found at: https://github.com/swirldev/swirl_courses#swirl-courses

There are instructions on how to install courses directly from swirl at the command line.

To load swirl:

```
library("swirl")
swirl()
```

You can install courses in swirl:

```
install_course("Course Name")
```

If this doesn't work, try:

```
install_from_swirl("Course Name")
```

To start a course, write `swirl()` in the command line. The program will guide you from there.

Course: R Programming

2017-01-05

2. Workspace and Files

R provides a common API for interacting with files and directories.

- `getwd()`: Get working directory

- `ls()`: List objects in local workspace
- `list.files()` or `dir()`: Lists files in wd
- `dir.create("Path to dir")`: Create directory in wd
- `setwd("path_to_dir")`: Sets the wd
- `file.create("path_to_file")`: Creates file in wd
- `file.exists("path_to_file")`: Checks to see if file exists
- `file.info("filename")`: Returns info about file. Can grab specific items from this using $
- `file.rename("old_name","new_name")`: Changes file name
- `file.remove("filename")`: Deletes file
- `file.copy("file_to_copy", "copy_name")`
- `file.path("filename")`: Provides path to file
- `dir.create("name", recursive=FALSE)`: Used to create directories. Can create nested recursive directories using the `recursive=TRUE` option.
  - **E.g.** `dir.create(file.path("testdir2", "testdir3"), recursive=TRUE)` creates nested directories.
- `unlink("directory", recursive=FALSE)`: Use to delete directories. Must have recursive=TRUE in order to work

`args(func)` returns the arguments of a function.
E.g. `args(list.files)`

7. Matrices and Data Frames

`dim()` allows you to get or set the dim attribute for an object in R.
`attributes()` allows us to check the attributes of an object
`length()` tells you the length of an object
`class()` tells you the class of an object

`cbind()` combines objects together

`data.frame()` creates a data frame

10. `lapply()` and `sapply()`

These programs are a concise way to implement the Split-Apply-Combine strategy for data analysis.
Each member of the `*apply()` functions will split up some data into smaller pieces, apply a function to each piece, and then combine the results.

`unique()` returns a vector with all duplicate elements remoed. Returns only unique elements.

12. Looking at Data

R has functions to look at data.

Data frames are the default class into which data is read in R.

`nrow()` tells you the number of rows
`ncol()` tells you the number of columns
`object.size()` tells you how much memory the object is taking up
`names()` will return a vector with the column names
`head()` shows you the top of a data set. By default, only first 6 rows. Can select how many rows to preview by setting the option `n=##`
`tail()` shows you the bottom of the data set
`summary()` gives you basic details about the data
`str()` also gives you a summary

Course: Data Analysis

1. Central Tendency

According to Wikipedia: "Data are values of qualitative or quantitative variables, belonging to a set of items". Often the set of items that we are interested in studying is referred to as the **population**. Data analysis usually involves studying a subset, or **sample**, or an entire population.

Data analysis should always start with a specific question of interest. E.g. "What percentage of people living in the US are over six feet tall?"

Here our population of interest is everyone living in the US. Since it's impractical to measure the heights of 300M people, we could instead choose 100 random people and measure their heights. Our hope would be that this sample of 100 people is **representative** of the entire US population.

The purpose of analyzing a sample is to draw conclusions about the population from which the sample was selected. This is called **inference** and is the primary goal of **inferential statistics**.

In order to make any inferences about the population, we first need to describe the sample. This is the primary goal of **descriptive statistics**.

If we want to describe our sample using just one number, how would we best do it? A good start is to find the center, the middle, or the most common element of our data. Statisticians call this the **central tendency**.

There are three methods for finding such a number and the applicability of each method depends on the situation. Those three methods are the **mean**, **median**, and **mode**.

The **arithmetic mean**, or simple the **mean** or **average**, is the most common measurement of central tendency.

Syntax:

```r
mean()
```

Extreme values in our dataset can have a significant influence on the mean. This can be misleading. An alternative to the mean, which is not influenced at all by extreme values, is the **median**. The median is computed by sorting all values from least to greatest and then selecting the middle value. If there is an even number of values, then there are actually two middle values. In this case, the median is equal to the mean of the two middle values.

Syntax:

```
median()
```

Finally, we may be most interested in finding the value that shows up the most in our dataset. In other words, what is the most common value? This is called the **mode** and it is found by counting the number of times that each value appears in the dataset and selecting the most frequent value.

2. Dispersion

3. Data Visualization