Introduction to R, Day 2

Communication Research Methods

Jennifer Pan

Assistant Professor
Department of Communication
Stanford University

January 15, 2016

Announcements

Announcements

- Pset 1: provide rationale for answers

Announcements

- Pset 1: provide rationale for answers
- Friday section room (see syllabus)

Overview

Overview

- Last time

Overview

- Last time
  - Installed R and RStudio

Overview

- Last time
  - Installed R and RStudio
  - Started using RStudio

Overview

- Last time
  - Installed R and RStudio
  - Started using RStudio
  - Learned how to create / manipulate objects

Overview

- Last time
  - Installed R and RStudio
  - Started using RStudio
  - Learned how to create / manipulate objects
- Today

Overview

- Last time
  - Installed R and RStudio
  - Started using RStudio
  - Learned how to create / manipulate objects
- Today
  - Review: creating / manipulating objects

Overview

- Last time
  - Installed R and RStudio
  - Started using RStudio
  - Learned how to create / manipulate objects
- Today
  - Review: creating / manipulating objects
  - Some useful functions

Overview

- Last time
  - Installed R and RStudio
  - Started using RStudio
  - Learned how to create / manipulate objects
- Today
  - Review: creating / manipulating objects
  - Some useful functions
  - Load datasets into R

Functions

- We have seen several functions: sqrt(), class(), c()
- Format: `funcname(input)`
  - `funcname`: function name
  - `(input)`: input object (also called arguments)
- Useful functions
  - `length()`: length of a vector or equivalently the number of its elements
  - `min()`: min value
  - `max()`: max value
  - `range()`: range of data
  - `mean()`: mean
  - `sum()`: sum all values in vector
  - `names()`: access and assign names to elements of a vector
- To avoid confusion and problems stemming from the order, specify name of argument

Data Files

- Vectors: manually entered data into R (not efficient)
- Most times: load data from an external file
- We will deal with two types of data
  - csv: comma separated values
  - RData: collection of R objects including datasets

Local Files: Change Working Directory

- Open RStudio
- Create a New R Script
- To load a file, you must know
  - where the file is on my computer (what is the file path)
  - copy the file path
  - change my working directory in RStudio to the directory where the file is
- Check current working directory in RStudio: `getwd()`
- Change current working directory in RStudio:

```
setwd("username/folder/folderwithdata/")
```

Local Files

- csv
  - use `read.csv()`
  - use assignment operator to save as an object
  - name of object is up to you
  - never change the name of the data files you use in this class
- RData
  - use `load()`
  - do not use assignment operator, R objects stored in the RData file already have object names

Learn about loaded data

- use `class()`
- often are `data.frame` objects: collection of vectors, but we can think of it like a spreadsheet
- useful functions
  - `names()`: vector of variable names
  - `nrow()`: number of rows
  - `ncol()`: number of columns
  - `dim()`: combines ncol and nrow
  - `summary()`: for each variable, the min, 25th percentile, median, 75th percentile, max
  - `View()`: same as clicking in Environment, shows data in table format
- to access an individual variables of data.frame (as a vector)
  - $ operator
  - indexing []
- missing values: `NA`, some function may need `na.rm=TRUE` to work

Saving Objects

- We can write objects are .csv or RData
  - For RData: `save(UNpop, file = "myUNpop.RData")`
  - For csv: `write.csv(UNpop, file = "myUNpop.csv")`

Packages

- R is open source
- Large community of people who contribute various functionalities as R packages
- Example: `foreign` package to read in data from programs like STATA and SPSS
  - The first time you use a package, you have to install it:
    `install.packages(‘‘foreign")`
  - Every time you use the package (with a new session of RStudio), you have to load it:
    `library(‘‘foreign")`
  - After package is loaded, you can use the functions to load "foreign" data:
    `read.dta("UNpop.dta")`