Introduction to R, Part I

Rebecca C. Steorts, STA 360

Agenda

- Reproducible Research
- Using R, RStudio, Markdown
- Simple Illustrations
- Assignment Operator
- Functions

Reproducible Research

Reproducible research is the idea that data analyses, and more generally, scientific claims, are published with their data and software code so that others may verify the findings and build upon them.

-Johns Hopkins, Coursera

The R Console

Basic interaction with R is by typing in the **console**, a.k.a. **terminal** or **command-line**

You type in commands, R gives back answers (or errors)

Menus and other graphical interfaces are extras built on top of the console

RStudio

- RStudio is very easy and simple to use. It can be downloaded from R Studio Download.
- RStudio is not R.
- RStudio mediates your interaction with R.

What is Markdown?

- Markdown is a lightweight markup language for creating HTML, PDF, or other documents.
- Markup languages are designed to produce documents from human readable text.
- This promotes research/materials that are reproducible.
- Also, RStudio integrates with LaTeX.

Why Markdown?

- It's easy to learn.
- It really pushes at reproducible code and documentation.
- Once this basics are down, you can do things that are more fancy.

Getting started with RStudio + Markdown

- A cheatsheet is given for simple markdown commands (R Markdown Cheat Sheet)
- Tysetting equations can be slightly different than LaTeX. There are some resources here! (LaTeX Typesetting)

Simple Illustration in RStudio

```
1+6
```

```
## [1] 7
```

```
x <- 4
(x + 2)
```

```
## [1] 6
```

```
set.seed(738)
```

Data in R

Most variables are created with the **assignment operator**, `<-` or `=`

```
average.rent.dollar <- 800
average.rent.dollar
```

```
## [1] 800
```

```
dollar.to.euro = 0.93
average.rent.dollar*dollar.to.euro
```

```
## [1] 744
```

The assignment operator also changes values:

```
average.rent.euro <- average.rent.dollar*dollar.to.euro
average.rent.euro
```

```
## [1] 744
```

```
average.rent.euro <- 744
average.rent.euro
```

```
## [1] 744
```

The workspace

What names have you defined values for?

```
ls()
```

```
## [1] "average.rent.dollar" "average.rent.euro"   "dollar.to.euro"
## [4] "x"
```

Getting rid of variables:

```
rm("average.rent.euro")
ls()
```

```
## [1] "average.rent.dollar" "dollar.to.euro"      "x"
```

What functions are available to you in the workspace:

The command **search** lists all packages that are currently loaded in your workspace

```
search()
```

```
## [1] ".GlobalEnv"        "package:stats"     "package:graphics"
## [4] "package:grDevices" "package:utils"     "package:datasets"
## [7] "package:methods"   "Autoloads"         "package:base"
```

The command **help** will give the same result, but allow this in an interactive form in your side window.

```
help(package = utils)
```

The command **?** allows you to delve into the help package regarding a specific function.

```
?hist()
```

Commonly used functions that you should review

1. ?length(), ?dim(), ?names(), ?ls(), ?class()
2. ?mean(), ?median(), ?var(), ?sd()
3. ?rnorm() (and other similar calls)
4. ?read.table, ?read.csv(), ?write.table(), ?write.csv()
5. ?hist(), ?plot(), ?density()
6. ?par(mfrow)
7. ?apply()
8. ?dplyr() (need to load the dplyr library)
9. ?matrix(), ?data.frame(), ?as.numeric()
10. rm(list = ls()) (This clears the R workspace)