The Very Basics of R

mf

November 3, 2017

Today's plan

- 1. Getting around in R studio
- 2. Wow!
- 3. Boring syntax stuff
- 4. Regressions

Why R? (R vs. STATA)

- R does a bunch of things better than STATA
  - Web stuff
  - Manipulating weird data formats
  - Graphics
  - Machine learning
  - Big Data
- R is much more commonly used by data scientists
  - (also Python)
- R is free
- However:
  - R is way harder to learn
  - R is much less practical for econometrics

Getting started with R Studio

- R studio is not R
  - Open Rx64 or Ri368 program (just to see it)
  - We'll never do this again. Just use R Studio.
- Open R studio
  - 4 Panes: Source (code) / Console (output) / 2 more for a bunch of things
  - Make a new project (File/NewProject) - set the directory to wherever u want
  - Open a new R script
  - Type "5" and hit control+Enter on the keyboard

```
# When I write "5" in the code and press Ctrl+Enter
5
```

```
## [1] 5
```

```
# is the output in the console
```

How R differs from Stata in usage

- R functions with "Work Spaces" associated with a project file .Rprj
- R holds many objects/datasets at once in the "environment"

```
data(cars); data(state)
```

- You usually run R programs line-by-line
- Everything depends on packages
  - kinda like ado files, but not exactly
  - packages contain a bunch of functions
    (not like ado file, which only contains one command)
  - tons of flexibility
  - all made by uncoordinated private users: very little consistency in syntax
  - often conflicts between packages
- You have to load packages every time into your "workspace"

Packages

```r
# You install packages with the command install.packages():
install.packages("dplyr")
# Then the packages exist in your local library (see panel)

# But they are not loaded until you run the library() command.
library(dplyr)
library(plyr)

# There are lots of competing/conflicting packages:
library(xlsx)
library(xlsx2)
```

The very basics of R syntax

```
# 1) Comments start with a Hashtag. No multiline comments.
# 2) Operations get resolved (i.e. you will see 16, not "8+8"
8+8
```

```
## [1] 16
```

```
sqrt(244)
```

```
## [1] 15.6205
```

```
# 3) Assignments are done with " <- " or " = "
a <- 4
b = 20
a + b
```

```
## [1] 24
```

The very basics of R syntax (2)

```
# 4) Commands figure out line breaks automatically
20 +
sqrt(
        4)
```

```
## [1] 22
```

```
# 5) But you can also use ";"
c <- 1; d <-2; e <-3 ;
c + d + e
```

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

The very basics of R syntax (3)

```
# 6) vectors are defined with the c() function
c(1,2,3,4)
```

```
## [1] 1 2 3 4
```

```
# 7) R thinks of most objects as matrices (or arrays)
c1 <- c(1,2,3,4)
c2 <- c(2,6,5,4)
cbind(c1,c2)
```

```
##      c1 c2
## [1,]  1  2
## [2,]  2  6
## [3,]  3  5
## [4,]  4  4
```

The very basics of R syntax (4)

```
# 8) You can mix data types (sortof)...
v <- c(1,3*2,3,"J. Bieber")
v
```

```
## [1] "1"         "6"         "3"         "J. Bieber"
```

```
# 9) Operators are mostly like stata
# ==    is logical equal
# !     means NOT
# &     means AND
# |     means OR
# (there are more)

# 10) We'll get back to syntax later.
```