SOC 4650/5650: Week 02 R Quick Reference

Christopher Prener, Ph.D.

January 24th, 2017

Read and Write .csv Data

Import `read.csv("data.csv", stringsAsFactors = FALSE)`[1]

Export `write.csv(dataFrame, "data.csv", na = "")`[2]

[1] The `stringsAsFactors = FALSE` option ensures that string data are preserved as string and not converted to numeric data.

[2] By using the `na = ""` option, blank cells will be used to represent missing data rather than inserting the text `N.A.`. This will ensure that numeric variables passed via `.csv` to ArcGIS will remain numeric.

List Data Frame Details

`str(dataFrame)`[3]

[3] Optionally, you can specify a variable name as well to restrict your output to a single variable.

List Variable Type

```
class(dataFrame$varName)
```

List Variable Descriptive Statistics

```
summary(dataFrame$varName)
```

Frequency Table

```
table(dataFrame$varName)
```

Tidy Output

Basic Syntax `tidy(outputSyntax)`[4]

e.g. `tidy(table(dataFrame$varName))`

[4] Function from the `broom` package, which is part of the `tidyverse`.

Saving Tidy Output `tidyObject <- tidy(outputSyntax)`

e.g. `tidyObject <- tidy(table(dataFrame$varName))`

Re-ordering Observations

Low-to-High `arrange(varName)`[5]

[5] Function from the `dplyr` package, which is part of the `tidyverse`.

High-to-Low `arrange(desc(varName))`[6]

[6] Functions from the `dplyr` package, which is part of the `tidyverse`.

Piping Functions

Pipe Operator `%>%` [7]

[7] Function automatically loaded from the `magrittr` package by the `dplyr` package.

Example

```
newDataFrame <-
  tidy(table(sourceDataFrame$varName)) %>%
  arrange(desc(Freq))
```

In this example, we create a frequency table for `varName` and arrange it as a "tidy" data frame. **Then** we re-order the observations in that data frame using the `Freq` from highest value to lowest value. When we say "**then**" when describing the process, we use the pipe operator. These operations are then assigned to the `newDataFrame`.

This seems complicated at first, but it is actually a more efficient way of writing code for `R`. Data frame names need only be specified when necessary, rather than multiple times in every line of code. The code itself is also easier to read and interpret.

Selecting Variables

`select(varList)`[8]

[8] Function from the `dplyr` package, which is part of the `tidyverse`.

Listing Observations

Basic Syntax `head(dataFrame, val)`

By default, `head(dataFrame)` will give you the top six observations for *all* variables. Specifying `head(dataFrame, 10)` will give you the first ten observations for *all* variables.

Specific Observations for Specific Variables

```
dataFrame %>%
  select(varName1, varName2, varName3) %>%
  head(10)
```