STAR511: R Hints and Help

Download R and RStudio

- To install R, go to www.r-project.org, and click on "download R", choose a CRAN mirror, and download R for your platform (binaries for base distribution).
- To install RStudio, go to www.rstudio.com. Select Products > Open Source > RStudio. We will use the free RStudio Desktop Open Source Edition.
- R is updated frequently, with a few major releases each year.

R Markdown/Saving Work

- For this class we will use RMarkdown for examples, HW and exams. This a convenient way to save code plus text.
- **We recommend knitting to Word**, as this generally causes fewer problems as compared to knitting to pdf.
- From RStudio, go to File > New File > R Markdown. Choose Document (default), provide a title and author and choose "Word". This provides a small mock document. Generally want to start by deleting initial text and code chunks.
- Use insert to insert new R code chunks.
- As you are working, run code within chunks and check output.
- Save .RMD file as you go and certainly save before exiting R!
- When all code chunks run without errors and you are happy with the result, knit to a document by selecting "Knit". The file will not knit if there are errors within the R code chunks.
- Two helpful code chunk options:
    - echo = FALSE prevents the **code** from appearing in the knitted document.
    - include = FALSE prevents the **output** from appearing in the knitted document.
    - The use of these options and multiple code chunks can go a long way in creating a very clean document.
- If you want to try knitting to pdf, first install and load the R package "tinytex".
- We do NOT recommend saving "workspace image" on exit. The idea is if we save our code (via R Markdown), we can easily recreate the results (objects and workspace).
- Other options for saving work include scripts, notebooks and projects. It is also possible to save individual objects. These approaches are not needed for the course, but may be helpful in other situations.

Knitting Problems

- **Knit frequently to avoid last minute problems.**
- To figure out where problems lie, it can be helpful to close R Studio, reopen and rerun the code chunks (in order).
- All R code chunks need to run without errors before the document will knit.

- Code to import required data needs to be included within the R markdown document (in a code chunk). Order matters: the import code needs to appear before the data is referenced.
- Code to load required packages needs to be included within the R markdown document (in a code chunk). Order matters: the package needs to be loaded before it is used.
- The use of install.packages() occasionally prevents the document from knitting. A package only needs to be installed once (the first time you use it). With this in mind install.packages() does not need to be included in the markdown document. Try removing it.
- If you get an error about "permission denied" it is likely that you are trying to re-knit an open document. Try closing any (Word, pdf) open versions.

Importing Data

- For this class we will use CSV files (comma separated values) and use read.csv() for importing.
- **Import Option #1:** Move data file into same folder with .RMD document. It should be the exact same folder, not a sub-folder. If you do this, you should only need to specify the file name.

```
InData <- read.csv("dataname.csv")
InData <- read.csv("dataname.csv",quote = "'")
```

- **Import Option #2**: Specify full filepath location

```
InData <- read.csv("filepath/dataname.csv")
InData <- read.csv("filepath/dataname.csv",quote = "'")
```

Replace "filepath" and "dataname" with file specific information.
    - Finding file path for Windows: Navigate to and select the file of interest. Hit shift and right click, choose "Copy as path…". Copy this into the read.csv code above. Note you will need replace \ (backslash) with / (forward slash).
    - Finding file path for Mac: Open finder window and browse to your .csv file. Right click on the file name which reveals a dropdown list for file options. Now hold the "option" button (next to the command button). Notice among the options is to copy the file as Pathname. Paste this into quotes for read.csv. Note: Mac users can shorten pathnames using "~". e.g. "~/Dropbox/STAT511/Assignments/Assign1/ex3-30.TXT"
- **Import Option #3**: Use RStudio Import
From RStudio, go to File > Import Dataset > From Text (base), navigate to file location, **check options** (including Name, Separator = Comma, Quote = Single quote (')) and hit Import. The read.csv code will appear in the Console. Important: you need to **copy this code into an R code chunk**, in order to knit the document.
- **Textbook Data Sets** are available from text book companion website. We recommend using the files in CSV format ("ASCII-comma"). These files can be imported using `read.csv("dataname.csv", quote = "'")` as shown above. The quote option is used because the column names in the original data are (single) quoted. If this option is omitted, you will end up with "funny" column names (ex: X.VarName.)

- Full list of options (skipping rows, comment characters, etc) are given in the help page for read.csv**.**
- Other options for importing: read.table() is a related option for importing. read_csv() is from the readr package. Excel files can be imported (and exported) to R. This can be done using the readxl or XLConnect packages. These approaches are not needed for the course, but may be helpful in other situations.

ALWAYS look at the data!

- From RStudio Environment tab, click on the blue button to the left of the data name. This will provide the number of observations (rows) and variables (columns). It will also provide the exact column names and variable types. This is equivalent using the code `str(InData)`**.**
- From RStudio Environment tab, click on the data name. This will open a new window showing the data. This is equivalent using the code `View(InData)`**.**

Specifying Columns

- $ can be used to reference a specific column from a data.frame.
  In other words, $ can be used to extract a vector from a data.frame.
  Ex: `mean(InData$Var)`
- Many (but not all) R functions have a data = option. This allows us to avoid $.
  When available, use the data = option!
  Ex: `boxplot(Y ~ X, data = InData)`
- Warning: R requires code to exactly match the column names in the data.frame. R is case sensitive.
- Other options include with() and attach().

Finding Help

- If you know the function name, from RStudio Help tab, search for the function.
  Scroll down to the bottom of the help page to see example code!
- If you want to do something, but don't know the function name, just google it! Ex: "R loess".
- If you want to work from a reference manual, R manuals are available r-project.org under Documentation, Manuals. Note the link for Contributed Documentation.

Installing and Loading Packages

- R comes with a standard set of packages, but many more (thousands!) are available for download. The good news is that this greatly extends R's functionality. The bad news is that code syntax is not always consistent across packages.
- The <u>first</u> time you use a package you need to <u>install</u> it.
  From the RStudio Packages tab, choose Install. Start to type the package name (case sensitive). Be sure the "Install dependencies" box is checked!

- Every time you want to use a package you need to load it.
  Within an R code chunk use library() or require() to load packages.
- **Best practice:** Include code to load all required packages in an early code chunk.
- The sessionInfo() function will give you information about R and package versions and which packages have been attached/loaded.

tidyverse

- tidyverse refers to a popular collection of R packages for data manipulation and graphing.
- These packages are not required for most course topics, but will sometimes be included in the examples.
- The good news about tidyverse is that it can be extremely helpful for more advanced analyses. The bad news about tidyverse is that the syntax may not be consistent with other (base R) functions and that functions are sometimes "retired".