

NC STATE UNIVERSITY

Intermediate Programming in R Part I

Justin Post
August 16-17, 2018

Course Schedule

Daily agenda:

- 9:30-10:40 Session
- 10-minute break
- 10:50-12:00 Session
- 12:00-1:15 Lunch
- 1:15-2:25 Session
- 10-minute break
- 2:35-3:45 Session

What do we want to be able to do?

- Communicate findings effectively
- Document findings
- Make process reproducible
- Share process

Where do we start?

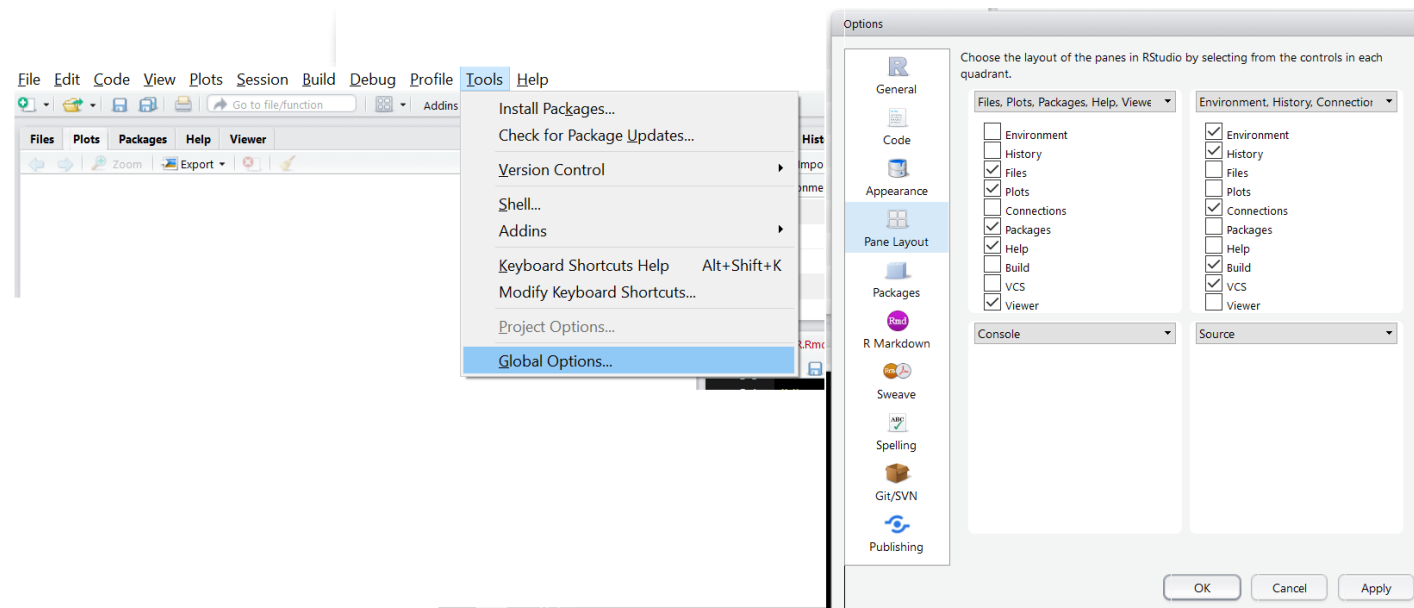
- RStudio and Background Skills
- R Markdown Basics
 - Code Chunks
 - Images/Equations/Misc.
- R Markdown Options
 - Documents: PDF, HTML
 - Presentations: Slides
 - Interactive Components
- R Shiny Applications/Presentations

RStudio

- Great integrated development environment (IDE)
- Four main 'areas' we'll use
 - Scripting and Viewing Area
 - Workspace/History
 - Files/Plots/Help
 - Console

RStudio

- Can rearrange panes



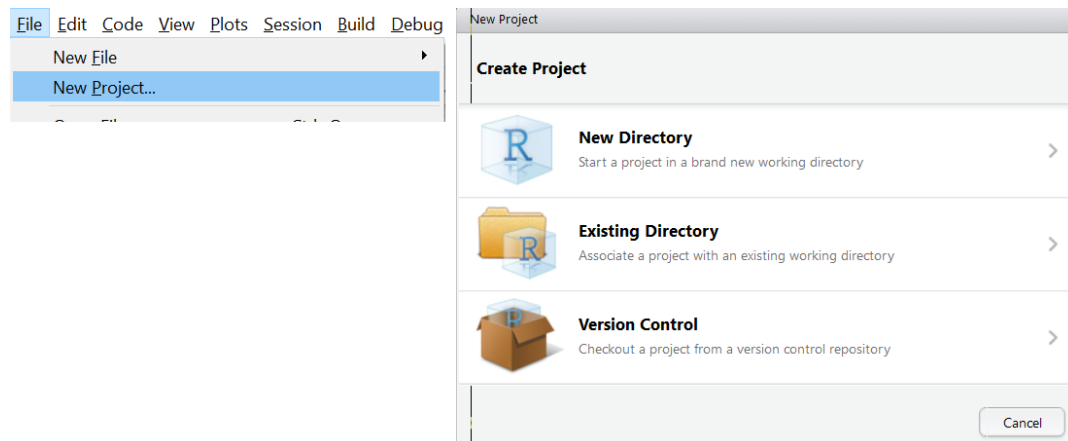
- Global options -> Appearance allows font/background changes
- Global options -> Code allows for soft-wrap of script files

RStudio - Project

- Often have many files associated with each analysis
- Keeping different undertakings separate can be difficult!
- Can use "Project" feature in R Studio
- Provides straightforward way to divide your work into multiple contexts. Each with their own:
 - Working directory
 - Workspace
 - History
 - Source documents

R Studio - Project

- Easy to create!



- Can save workspace, etc. and pick up right where you left off!
- Let's create one for today!
- Place the 'TextActivityPlainText.Rmd' file in the project folder

Background R Material

Packages - Many ways to accomplish the same thing in R

- How to choose?
 - Want 'fast' code
 - Want 'easy' syntax
 - Good default settings on functions
- Base R has reasonable defaults and syntax but functions are slower and aren't consistent
- "[TidyVerse](#)" - collection of R packages that share common philosophies and are designed to work together!
 - Very efficient code
 - Common syntax

Background R Material

- If not installed (downloaded) on computer

Background R Material

- Once installed, `library()` or `require()` to load

Tidyverse Syntax

- All packages have similar syntax! All work on **tibbles** (special data frames)
- Convert any data frame (or matrix) to a tibble using `tbl_df()`
- Nice printing properties (can sometimes cause issues though)
- Most packages have syntax: `function(data.frame, options)`

Tidyverse Syntax

- All packages have similar syntax! All work on **tibbles** (special data frames)
- Convert any data frame (or matrix) to a tibble using `tbl_df()`
- Nice printing properties (can sometimes cause issues though)
- Syntax: `function(data.frame, options)`
- Examples:

Background R Material

Data Frames

- Best R object for data sets
- Collection (list) of vectors of the same **length**

```
## # A tibble: 150 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##         <dbl>      <dbl>      <dbl>      <dbl> <fct>
## 1         5.1        3.5        1.4        0.2 setosa
## 2         4.9         3         1.4        0.2 setosa
## 3         4.7        3.2        1.3        0.2 setosa
## 4         4.6        3.1        1.5        0.2 setosa
## 5          5         3.6        1.4        0.2 setosa
## # ... with 145 more rows
```

Background R Material

Data Frames

- Accessing elements: multiple ways

```
## # A tibble: 4 x 3
##   Sepal.Width Petal.Length Petal.Width
##   <dbl>       <dbl>       <dbl>
## 1     3.5         1.4         0.2
## 2     3           1.4         0.2
## 3     3.2         1.3         0.2
## 4     3.1         1.5         0.2
```

Background R Material

Data Frames

- Accessing elements: multiple ways

```
## # A tibble: 1 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           <dbl>       <dbl>       <dbl>       <dbl> <fct>
## 1           5.1         3.5         1.4         0.2 setosa
```


Background R Material

Data Frames

- Accessing elements: multiple ways

```
## [1] 5.1 4.9 4.7 4.6 5.0 5.4 4.6 5.0 4.4 4.9 5.4 4.8 4.8 4.3 5.8 5.7 5.4
## [18] 5.1 5.7 5.1 5.4 5.1 4.6 5.1 4.8 5.0 5.0 5.2 5.2 4.7 4.8 5.4 5.2 5.5
## [35] 4.9 5.0 5.5 4.9 4.4 5.1 5.0 4.5 4.4 5.0 5.1 4.8 5.1 4.6 5.3 5.0 7.0
## [52] 6.4 6.9 5.5 6.5 5.7 6.3 4.9 6.6 5.2 5.0 5.9 6.0 6.1 5.6 6.7 5.6 5.8
## [69] 6.2 5.6 5.9 6.1 6.3 6.1 6.4 6.6 6.8 6.7 6.0 5.7 5.5 5.5 5.8 6.0 5.4
## [86] 6.0 6.7 6.3 5.6 5.5 5.5 6.1 5.8 5.0 5.6 5.7 5.7 6.2 5.1 5.7 6.3 5.8
## [103] 7.1 6.3 6.5 7.6 4.9 7.3 6.7 7.2 6.5 6.4 6.8 5.7 5.8 6.4 6.5 7.7 7.7
## [120] 6.0 6.9 5.6 7.7 6.3 6.7 7.2 6.2 6.1 6.4 7.2 7.4 7.9 6.4 6.3 6.1 7.7
## [137] 6.3 6.4 6.0 6.9 6.7 6.9 5.8 6.8 6.7 6.7 6.3 6.5 6.2 5.9
```

Background R Material

Data Frames

- Accessing elements: multiple ways

```
## # A tibble: 150 x 2
##   Sepal.Length Species
##         <dbl> <fct>
## 1         5.1 setosa
## 2         4.9 setosa
## 3         4.7 setosa
## 4         4.6 setosa
## 5          5  setosa
## # ... with 145 more rows
```

Background R Material

- Read in most any type of data with packages
 - readr (.csv, delimited data)
 - readxl (.xls, .xlsx)
 - haven (.sav, .dta, .sas7bdat)

Background R Material

```
## # A tibble: 3,141 x 20
##   region county state   msa  pmsa pop.density   pop pop.change age6574
##   <chr>   <chr>   <chr> <int> <int>      <int> <int>      <dbl>   <dbl>
## 1 South Autau~ AL    5240   NA        61 34222      11.9    5.70
## 2 South Baldw~ AL    5160   NA        67 98280      35.4    9.20
## 3 South Barbo~ AL      NA   NA        29 25417        2     8.20
## 4 South Bibb   AL      NA   NA        28 16576       9.20    6.70
## 5 South Blount AL    1000   NA        62 39248      10.6    7.40
## # ... with 3,136 more rows, and 11 more variables: age75 <dbl>,
## #   crime <int>, college <dbl>, income <int>, farm <dbl>, democrat <dbl>,
## #   republican <dbl>, Perot <dbl>, white <dbl>, black <dbl>, turnout <dbl>
```

Background R Material

Piping or Chaining

- Applying multiple functions: nesting hard to parse!
- Piping or Chaining with `%>%` operator helps

Background R Material

Piping or Chaining

- Applying multiple functions: nesting hard to parse!
- Piping or Chaining with `%>%` operator helps
- If `dplyr` or `magrittr` package loaded, can use anywhere

Background R Material

Piping or Chaining

```
## # A tibble: 11 x 17
##   county   msa pop.density   pop pop.change age6574 age75 crime college
##   <chr> <int>      <int> <int>      <dbl> <dbl> <dbl> <int>  <dbl>
## 1 Polk      NA         63  14416      14.7  13.7  10.9  1802   20.1
## 2 New H~  9200         643 120284      23.5   7.80  4.70  9778   21.2
## 3 Dare      NA         62  22746      77.4   8.30  4.10  8315   21.4
## 4 Pitt    3150         173 107924      24.7    6    3.90  4214   21.9
## 5 Forsy~  3120         661 265878      11.2   7.10  5.20  7976   24.1
## 6 Guilf~  3120         550 347420      12.8    7    4.90  7990   24.8
## 7 Watau~    NA         122  36952       20    6.30  4.30  2862   27.4
## 8 Meckl~  1520        1020 511433       33    5.70  3.70 11154   28.3
## 9 Durham  6640         647 181835      23.5   6.10  4.60  8375   33.4
## 10 Wake   6640         548 423380      51.7   4.70  3.10  6057   35.3
## 11 Orange 6640         249  93851      29.4   5.10  3.60  6848   46.1
## # ... with 8 more variables: income <int>, farm <dbl>, democrat <dbl>,
## #   republican <dbl>, Perot <dbl>, white <dbl>, black <dbl>, turnout <dbl>
```

Background R Material

Plotting

- R great for plotting
- We'll use `ggplot2` in `tidyverse`! [cheatsheet](#)
- Needs: Data Frame
- Aesthetic (`aes`) - maps variables to properties of geom
 - Ex: size, color, and x, y location(s)
- Geom layer(s) (visualization type(s))
- Coordinate system (mostly use Cartesian plane)
- Optional: Stat layer, titles, etc.

Background R Material

`ggplot2` needs and syntax

Needs:

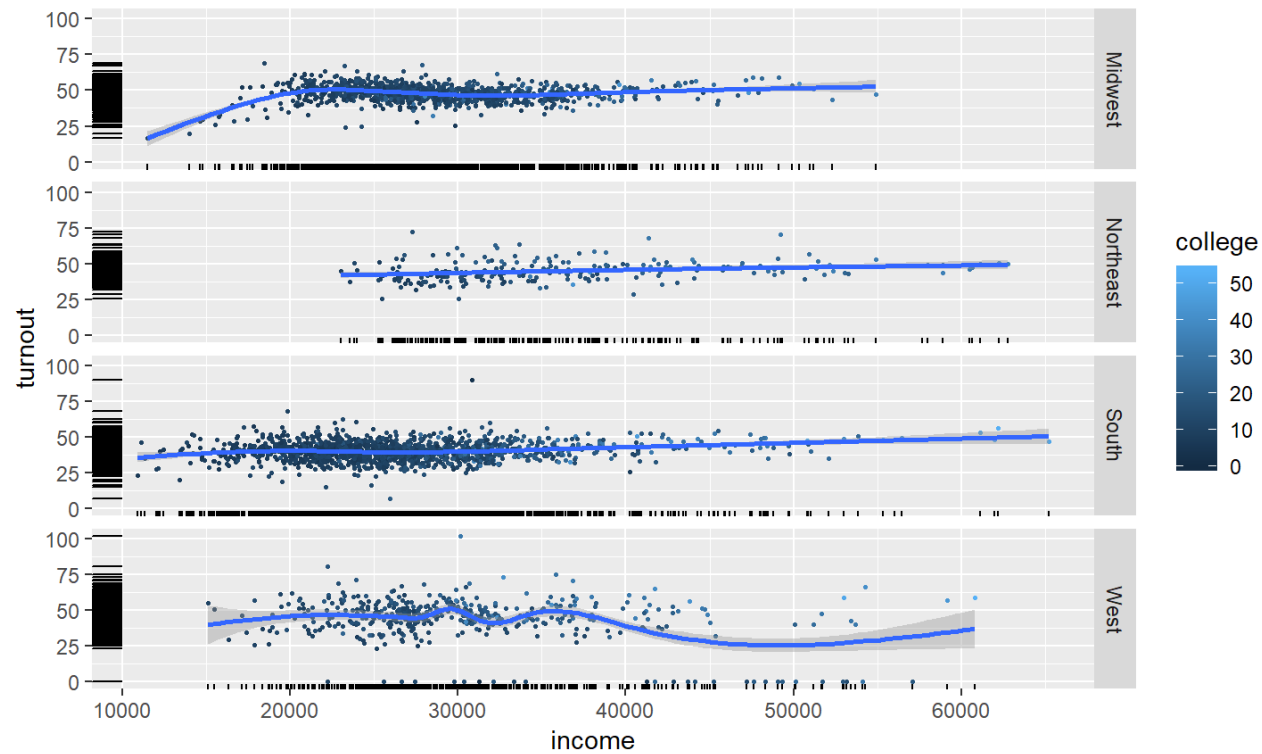
- Data Frame
- Aesthetic (aes) - maps variables to properties of geom
- Geom layer(s) (visualization type(s))
- Optional: Stat layer, titles, etc.
- Syntax:

Background R Material

- Settings that depend on a variable go in `aes`

```
g <- ggplot(votingData, aes(x = income, y = turnout))  
g + geom_point(size = 0.5, aes(color = college)) +  
  geom_smooth() +  
  geom_rug() +  
  facet_grid(region ~ .)
```

Background R Material



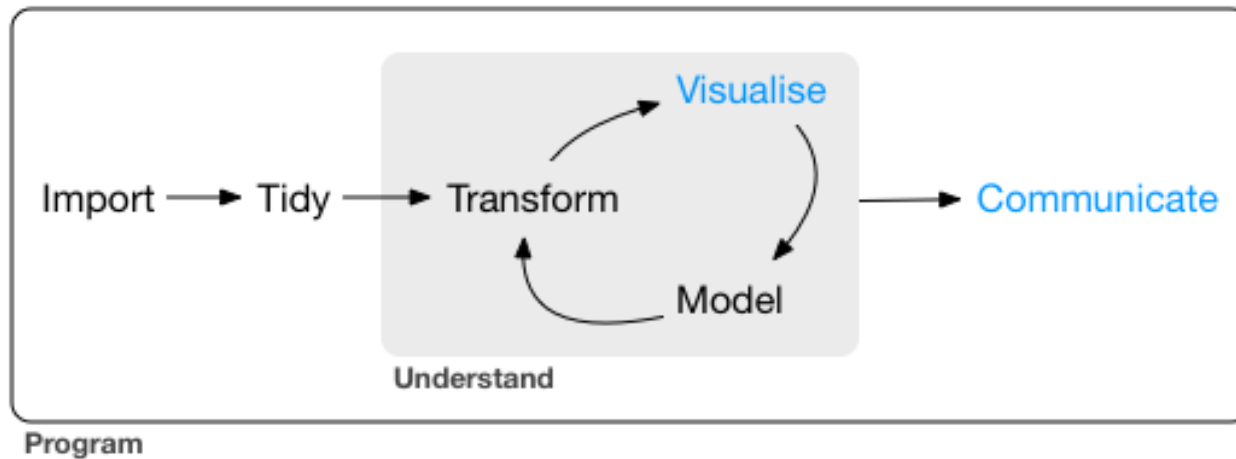
What do we want to be able to do?

- Communicate findings effectively
- Document findings
- Make reproducible process
- Share process

Where do we start?

- Review of Key Concepts
- R Markdown Basics
 - Code Chunks
 - Images/Equations/Misc.
- R Markdown Options
 - Documents: PDF, HTML
 - Presentations: Slides
 - Interactive Components
- R Shiny Applications/Presentations

R Markdown Basics



(From R for Data Science)

R Markdown Basics

- Can read data into R
- Ability to manipulate it
- Likely know best ways to model and visualize it
- Doesn't matter how great your analysis is unless you can explain it to others :)
- Need to communicate results effectively

R Markdown Basics

What is Markdown?

- Formatting syntax for authoring HTML, PDF, slide shows, books, and more.
- Digital "Notebook": Program that weaves word processing and code. [Example](#)
- Can do interactive documents!
- [Intro video](#)

R Markdown Basics

How to use Markdown?

Designed to be used in three ways (R for Data Science)

- Communicating to decision makers (focus on conclusions not code)
- Collaborating with other data scientists (including future you!)
- As environment to do data science (documents what you did and what you were thinking)
- [Examples](#) of markdown documents

R Markdown Basics

Verbage

- Most have heard of HTML (HyperText Mark-up Language)
 - Write plain text that the browser interprets and renders

R Markdown Basics

Verbage

- Most have heard of HTML (HyperText Mark-up Language)
 - Write plain text that the browser interprets and renders
- Markdown is a specific markup language
 - Easier syntax
 - Not as powerful
- Any plain text file with .Rmd extension can be used

R Markdown Basics

R Markdown Basics

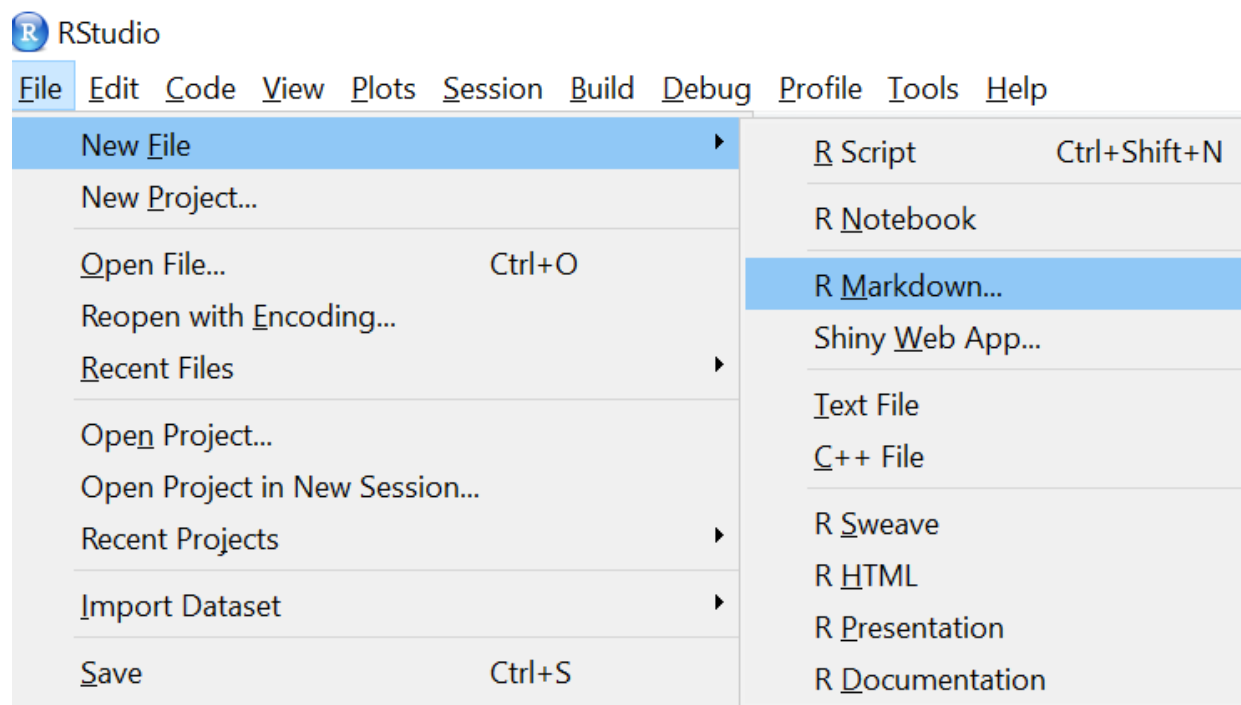
R Markdown file contains three important types of content:

1. (Optional) YAML header surrounded by `---`
2. Chunks of R code surrounded by `` ```
3. Text mixed with simple text formatting like `#` heading and

R Markdown Basics

Creating an R Markdown Document

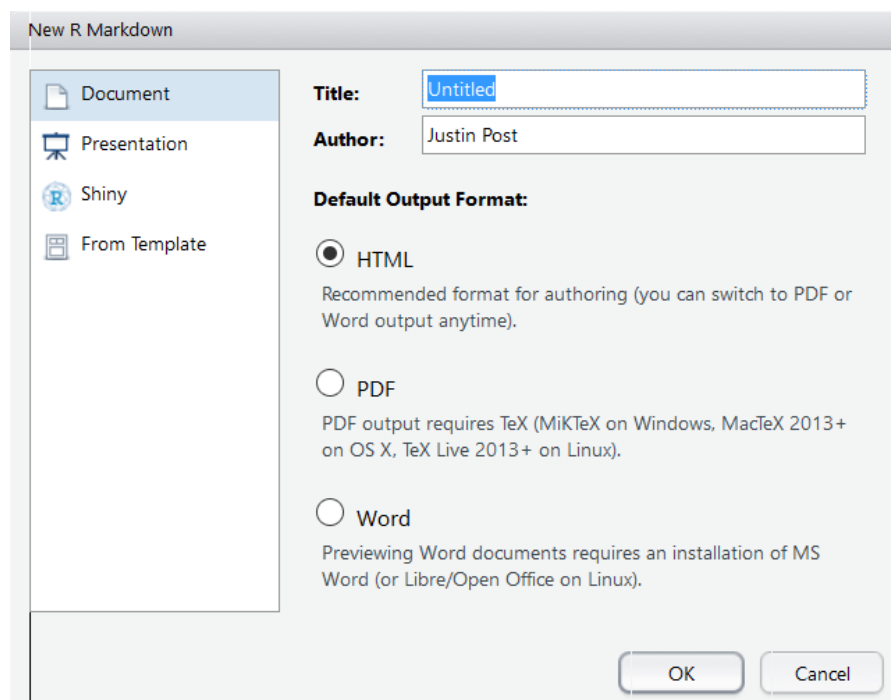
- R Studio makes it easy!



R Markdown Basics

Creating an R Markdown Document

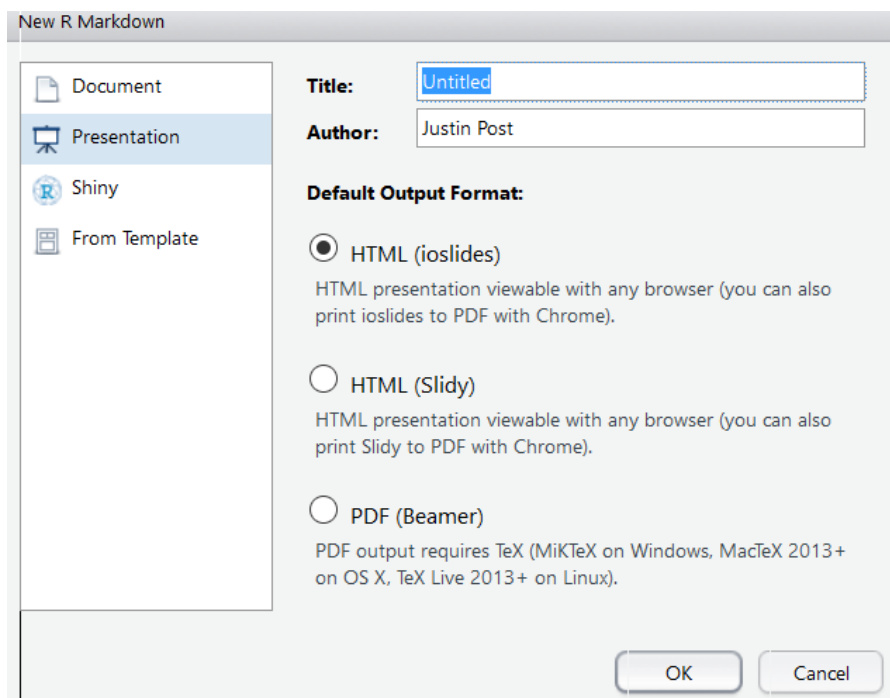
- Commonly used document types can be created



R Markdown Basics

Creating an R Markdown Document

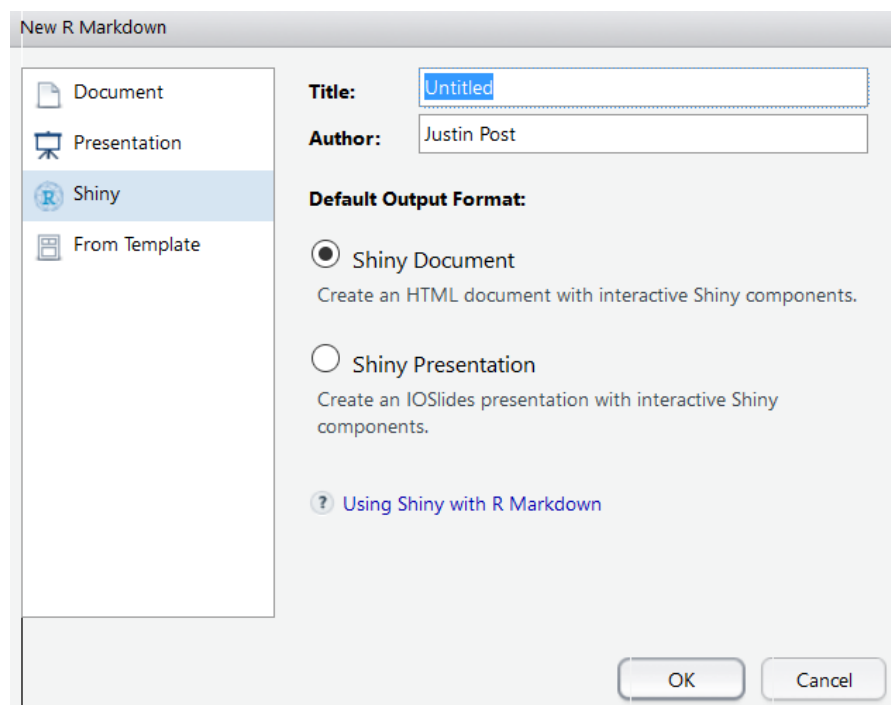
- Slide presentations



R Markdown Basics

Creating an R Markdown Document

- Truly Interactive Documents/Pages (requires R backend)



R Markdown Basics

- Create an HTML Markdown document!

```
---  
title: "Untitled"  
author: "Justin Post"  
date: "August 10, 2017"  
output: html_document  
---
```

- Top section: YAML header
- Define settings for document
- Author, Title, etc.
- Output type/Options

R Markdown Basics

- Below YAML header: 'r chunk'

```
```{r ggplot,eval=FALSE}  
select(iris, Sepal.Width)
ggplot(iris, aes(x = Sepal.Width, y = Sepal.Length)) +
geom_point()
```
```

- Start code chunk by typing ````{r} out or with CTRL/CMD + Alt + I
- Code will be executed when document is created
- Can specify options on individual code chunks

R Markdown Basics

- Below code chunk is plain text with markdown syntax

```
## R Markdown
```

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

- When file created, "##" becomes a header, "<...>" a link, and **Knit** bold font

R Markdown Basics

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document.

Where do we go from here?

- Figure out markdown syntax
- Look at "Notebook" feature
- Check options for code chunks
- Automate some things
- Change type of output
- Work with interactivity (shiny)

R Markdown Syntax Can Include...

- Plain text
- End a line with two spaces to start a new paragraph
 - Line breaks are not always added when you return!
 - Two spaces and a return drop marked up text down.
 - Can specify `
` as a line break
- **italics** `_italics_`
- ****bold**** and `__bold__`
- superscript² becomes superscript²
- ~~~~strikethrough~~~~ becomes ~~strikethrough~~
- Modify text in your .Rmd file and click Knit -> Knit to HTML (at the top)

R Markdown Syntax

- `[link](https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf)` becomes [link](https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf)
- `# Header 1` becomes a large font header
- `## Header 2` becomes a slightly smaller font header
- Goes to 6 headers
- Use of headers can automatically create a Table of Contents!
- Include an image: ``
- ``code`` becomes `code`
- Modify text in your .Rmd file and click Knit -> Knit to HTML (at the top)

R Markdown Syntax

- Can do lists: be sure to end each line with two spaces!
- Indent sub lists two spaces (I often do four for both)

* unordered list

* item 2

+ sub-item 1

+ sub-item 2

1. ordered list

2. item 2

+ sub-item 1

+ sub-item 2

• unordered list

• item 2

- sub-item 1

- sub-item 2

1. ordered list

2. item 2

• sub-item 1

• sub-item 2

R Markdown Syntax

- Can include nice tables

```
Table Header | Second Header | Col 3
----- | ----- | -----
Table Cell | Cell (1, 2) | Cell (1, 3)
Cell (2, 1) | Cell (2, 2) | Cell (2, 3)
```

| Table Header | Second Header | Col 3 |
|--------------|---------------|-------------|
| Table Cell | Cell (1, 2) | Cell (1, 3) |
| Cell (2, 1) | Cell (2, 2) | Cell (2, 3) |

- Add a list and a table in your .Rmd file and click Knit -> Knit to HTML (at the top)

Activity

- [Formatting Text Activity instructions](#) available on web
- Work in small groups
- Ask questions! TAs and I will float about the room
- Feel free to ask questions about anything you didn't understand as well!

What do we want to be able to do?

- Communicate findings effectively
- Document findings
- Make process reproducible
- Share process

Where are we at?

- Review of Key Concepts
- R Markdown Basics
 - **Code Chunks**
 - **Images/Equations/Misc.**
- R Markdown Options
 - Documents: PDF, HTML
 - Presentations: Slides
 - Interactive Components
- R Shiny Applications/Presentations

Code Chunks

We've already seen how to include an R code chunk:

```
```{r ggplot,eval=FALSE}  
select(iris, Sepal.Width)
ggplot(iris, aes(x = Sepal.Width, y = Sepal.Length)) +
geom_point()
```
```

- Add chunk via typing

```
```{r}  
code
```
```

- or **Ctrl/Cmd + Alt + I**
- Any R code can go into the chunk
- Code can be added in line: Ex: Iris has 150 observations
- Added by beginning with back-tick `r` and ending with a back-tick: Iris has ``r
length(iris$Sepal.Length)``

Notebook Functionality

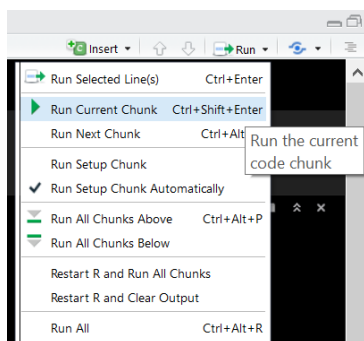
Data science notebook - virtual notebook environment used for literate programming

- Pairs the functionality of word processing software with a programming language
- Rendered markdown document captures R code and process
- R Markdown brings together the console and the script editor too!
- Blurs line between interactive exploration and long-term code capture.

Notebook Functionality

Within a chunk:

- Execute code with `Cmd/Ctrl + Shift + Enter` or with "Run"



- Results show up in editor!

Notebook Functionality

- Allows for quick iteration within a chunk: editing and re-executing - when you are happy, you move on and start a new chunk.
- Can run all code chunks with **Ctrl/Cmd + Alt + R**
- Can develop code and record your thoughts - similar to classic lab notebook in the physical sciences
- Go back to markdown template document, execute code chunk in-line and all code chunks. Add an inline code piece.

Back to Code Chunks

- Many options depending on chunk purpose!
- Can hide/show code with `echo = FALSE/TRUE`
- Can choose if code is evaluated with `eval = TRUE/FALSE`
- `Include = FALSE` is equivalent to `echo = FALSE, eval = TRUE`
- `message = TRUE/FALSE` and `warning = TRUE/FALSE` can turn on/off displaying messages/warnings
- `error = TRUE` allows file to be created with code that has an error

Back to Code Chunks

- Options useful for set-up code (usually first chunk after YAML header)
- Can set global options for all chunks
- Allows for easy change of audience!

Back to Code Chunks

- Options useful for set-up code (usually first chunk after YAML header)
- Can set global options for all chunks
- Allows for easy change of audience!

For a higher up

```
opts_chunk$set(echo = FALSE, eval = TRUE, warning = FALSE)
```

For a collaborator

```
opts_chunk$set(echo = TRUE, eval = TRUE, warning = FALSE)
```

Code Chunks

Can name code chunks to help organization!

- When calling a chunk, add name after `r`

```
` `` {r name-of-chunk, options...}  
code  
` ``
```

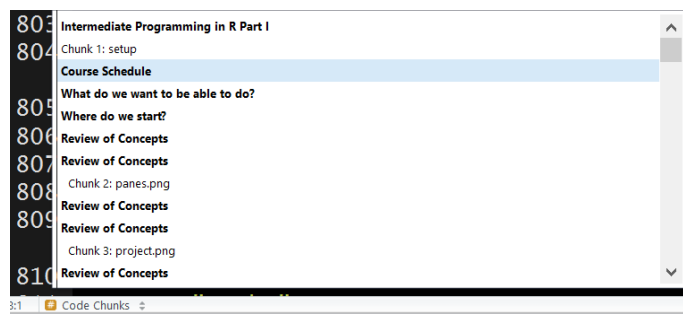
Code Chunks

Can name code chunks to help organization!

- When calling a chunk, add name after `r`

```
`r` {r name-of-chunk, options...}  
code  
`r`
```

- TOC type menu in bottom left of notebook editor!



Code Chunks

In a large analysis it may take a long time to run code chunks/knit your document

- Can "Cache" results! Code will only rerun if it has changed.
- Option to set up code dependencies using chunk names
- Use `cache = TRUE` in code chunk definition
- Can do global option for caching!

Code Chunks

In a large analysis it may take a long time to run code chunks/knit your document

- Can "Cache" results! Code will only rerun if it has changed.
- Option to set up code dependencies using chunk names
- Use `cache = TRUE` in code chunk definition
- Can do global option for caching!

```
opts_chunk$set(cache = TRUE)
```

- Delete folders created to rerun everything
- Modify `knitr::opts_chunk$set` in `setup` R chunk after your YAML header. Make changes and see differences! (Local settings overwrite global.) (Add `library(tidyverse)` to a code chunk to have messages that would pop up!)

Images/Equations and Misc.

Adding images in markdown: ``

- Not ideal... difficult to control size/scale
- Better way to add images use R code!
- `knitr` package has `include_graphics` function
- Use `knitr` or code chunk options to control size/scale!
- Ex:

```
````{r graphics, out.width = "800px", echo = FALSE}  
knitr::include_graphics(path/to/file)
````
```


Images/Equations and Misc.

Adding Equations

- Inline equation: `$A = \pi*r^{2}$` becomes $A = \pi * r^2$
- Block equation `$$A = \pi*r^{2}$$` becomes

$$A = \pi * r^2$$

- Outputting equations for HTML is done through MathJax (javascript)
- For PDFs it is done through LaTeX (may need to install)

Images/Equations and Misc.

Outputting data tables better with `kable` from `knitr` package

`summary(cars)`

```
##      speed      dist
##  Min.   : 4.0    Min.   :  2.00
## 1st Qu.:12.0    1st Qu.: 26.00
##  Median:15.0    Median : 36.00
##   Mean  :15.4    Mean   : 42.98
## 3rd Qu.:19.0    3rd Qu.: 56.00
##   Max.  :25.0    Max.    :120.00
```

`kable(summary(cars))`

| speed | dist |
|--------------|----------------|
| Min. : 4.0 | Min. : 2.00 |
| 1st Qu.:12.0 | 1st Qu.: 26.00 |
| Median :15.0 | Median : 36.00 |
| Mean :15.4 | Mean : 42.98 |
| 3rd Qu.:19.0 | 3rd Qu.: 56.00 |
| Max. :25.0 | Max. :120.00 |

`</div >`

Activity

- [Using Notebook Activity instructions](#) available on web
- Work in small groups
- Ask questions! TAs and I will float about the room
- Feel free to ask questions about anything you didn't understand as well!

What do we want to be able to do?

- Communicate findings effectively
- Document findings
- Make process reproducible
- Share process

Where are we at?

- Review of Key Concepts
- R Markdown Basics
 - Code Chunks
 - Images/Equations/Misc.
- R Markdown Options
 - Documents: PDF, HTML
 - Presentations: Slides
 - Interactive Components
- R Shiny Applications/Presentations

Common Outputs

R Markdown really flexible!



Common Outputs

Change output type in the YAML header:

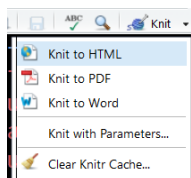
- HTML (a web page)

```
output: html_document
```

Use code explicity:

```
rmarkdown::render("file.Rmd", output_format = "word_document")
```

Use Knit menu:



Common Outputs

For HTML can include Table of Contents with options

```
output:  
  html_document:  
    toc: true  
    toc_float: true
```


Common Outputs

For HTML can include Table of Contents with options

```
output:  
  html_document:  
    toc: true  
    toc_float: true
```

For html_documents another option is to make the code chunks hidden by default, but visible with a click:

```
output:  
  html_document:  
    code_folding: hide
```

Common Outputs

- PDF
- May want to install LaTeX for equations

output: pdf_document

Common Outputs

- PDF
- May want to install LaTeX for equations

output: pdf_document

- Word

output: word_document

- Go back to template, change output type and knit. Add [some HTML options](#) and [PDF options](#) (careful to drop down lines and use spacing as done in documentation).

Common Outputs

Presentations/Slides (new slides with ##)

- `output: ioslides_presentation` - HTML presentation
- `slidy_presentation` - HTML presentation
- `beamer_presentation` - PDF presentation with LaTeX Beamer

Common Outputs

Presentations/Slides (new slides with ##)

- `output: ioslides_presentation` - HTML presentation
- `slidy_presentation` - HTML presentation
- `beamer_presentation` - PDF presentation with LaTeX Beamer
- Shiny (covered later) slides

```
runtime: shiny  
output: html_document
```

or

```
runtime: shiny  
output: ioslides_presentation
```

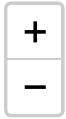
Interactivity

HTML documents inherently interactive

- Widgets can be included

```
library(leaflet)
leaflet() %>%
  setView(174.764, -36.877, zoom = 16) %>%
  addTiles() %>%
  addMarkers(174.764, -36.877, popup = "Maungawhau")
```

Interactivity



[Leaflet](#)

Interactivity

Interactive tables with DT library

```
library(DT)  
datatable(iris)
```


Interactivity

Interactivity

- 3d scatterplots with `rthreejs` package

Interactivity

Interactivity

Previous interactivity happened in the browser

- Great because anyone can access with a browser
- Bad because you can't have as much functionality as you want...
- Shiny allows for interactivity with R!
- Only con: Requires R running somewhere
- Examples: [Shiny Showcase](#), [Shiny Gallery](#)

What do we want to be able to do?

- Communicate findings effectively
- Document findings
- Make process reproducible
- Share process

Where are we at?

- Review of Key Concepts
- R Markdown Basics
 - Code Chunks
 - Images/Equations/Misc.
- R Markdown Options
 - Documents: PDF, HTML
 - Presentations: Slides
 - Interactive Components
- R Shiny Applications/Presentations

Activity

- [Outputs and Interactivity Activity instructions](#) available on web
- Work in small groups
- Ask questions! TAs and I will float about the room
- Feel free to ask questions about anything you didn't understand as well!

Parameters (Advanced)

Parameters can be added to the YAML header

- Can help to automate reports!

```
title: "NFL Reports"
author: "Justin Post"
date: "August 2018"
output: html_document
params:
  team: "Pittsburgh Steelers"
```

- Access via `params$team`
- Can 'Knit with parameters'
- Example: Let's open up the [NFL.Rmd document](#)

Parameters (Advanced)

- Create data frame for each class (here team)

```
scoreData <- read_csv("https://github.com/jbpost2/  
IntermediateR/blob/master/datasets/scoresFull.csv?raw=true")
```

```
reports <- tibble(  
  teamIDs = unique(scoreData$awayTeam),  
  filename = stringr::str_c("TeamID-", teamIDs, ".html"),  
  params = purrr::map(teamIDs, ~ list(team = .))  
)
```

Parameters (Advanced)

```
## # A tibble: 32 x 3
##   teamIDs          filename          params
##   <chr>          <chr>          <list>
## 1 San Francisco 49ers TeamID-San Francisco 49ers.html <list [1]>
## 2 Minnesota Vikings TeamID-Minnesota Vikings.html <list [1]>
## 3 New Orleans Saints TeamID-New Orleans Saints.html <list [1]>
## 4 New York Jets      TeamID-New York Jets.html <list [1]>
## 5 Arizona Cardinals TeamID-Arizona Cardinals.html <list [1]>
## # ... with 27 more rows
```

Parameters (Advanced)

Now knit via the following code:

```
reports %>%  
  select(output_file = filename, params) %>%  
  purrr::pwalk(rmarkdown::render, input = "NFL.Rmd")
```