R Basics

Justin Post

Our Tasks as a Data Scientist

- Read raw data in or connect to a database
- Manipulate data as need
  - Subset
  - Create new variables
- Summarize data to create meaningful insights
- Modeling data to make inference or predict outcomes
- Communicate our results via dashboards, documents, model files, etc.

R is a great language for all of these!

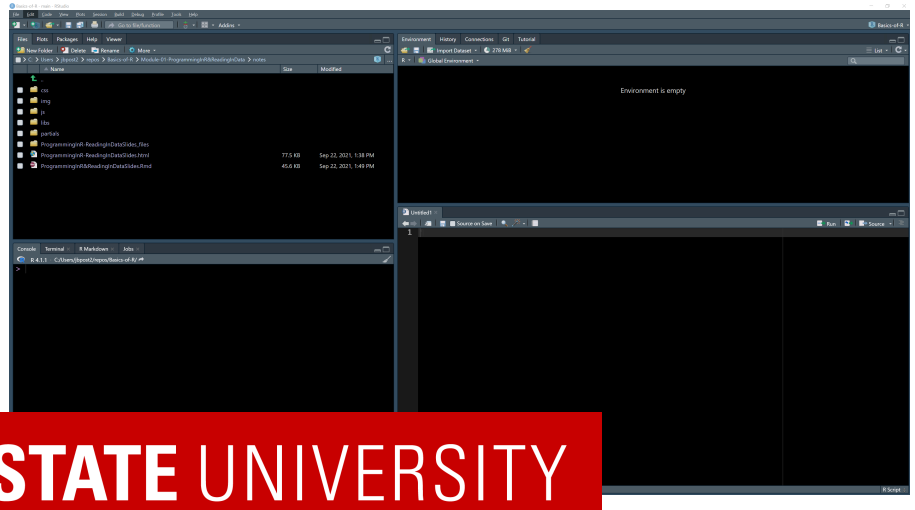
R vs RStudio

- R is our programming language
- Code in R through RStudio

RStudio IDE

In RStudio, four main locations

- Console (& Terminal)
- Scripting and Viewing Window
- Files/Plots/Packages/Help
- Environment (& Connections/Git)



NC STATE UNIVERSITY

Console

- Type code directly into the **console** for evaluation

```
#simple math operations
# <-- is a comment - code not evaluated
3 + 7
```

```
## [1] 10
```

```
10 * exp(3) #exp is exponential function
```

```
## [1] 200.8554
```

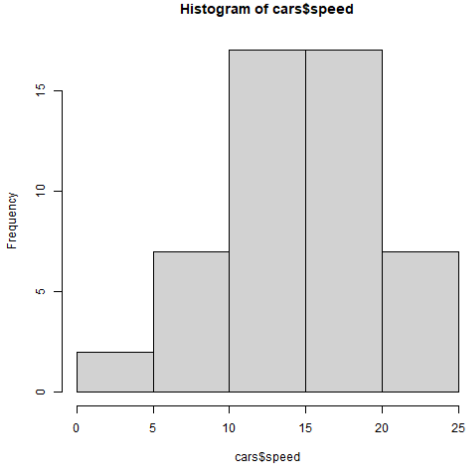
```
log(pi^2) #log is natural log by default
```

```
## [1] 2.28946
```

```
mean(cars$speed)
```

```
## [1] 15.4
```

```
hist(cars$speed)
```



Scripting and Viewing Window

- Usually want to keep code for later use!
- Write code in a 'script' and save script (or use markdown/quarto!)

Scripting and Viewing Window

- Usually want to keep code for later use!
- Write code in a 'script' and save script (or use markdown!)
- From script can send code to console via:
    - "Run" button (runs current line)
    - CTRL+Enter (PC) or Command+Enter (MAC)
    - Highlight section and do above

Files/Plots/Packages/Help

- Files (navigate through files)
- Created plots stored in `Plots` tab
    - Cycle through past plots\
    - Easily save
- Packages (update and install)
- Documentation within RStudio via `help(...)`
    - Ex: `help(seq)`

Environment

- We store **data/info/function/etc.** in R objects
- Create an R object via `<-` (recommended) or `=`

```
#save for later
avg <- (5 + 7 + 6) / 3
#call avg object
avg
```

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

```
#strings (text) can be saved as well
words <- c("Hello there!", "How are you?")
words
```

```
## [1] "Hello there!" "How are you?"
```

Environment

- Built-in objects exist like `letters` and `cars` don't show automatically

```
letters
```

```
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l" "m" "n" "o" "p" "q" "r" "s"
## [20] "t" "u" "v" "w" "x" "y" "z"
```

```
head(cars, n = 3)
```

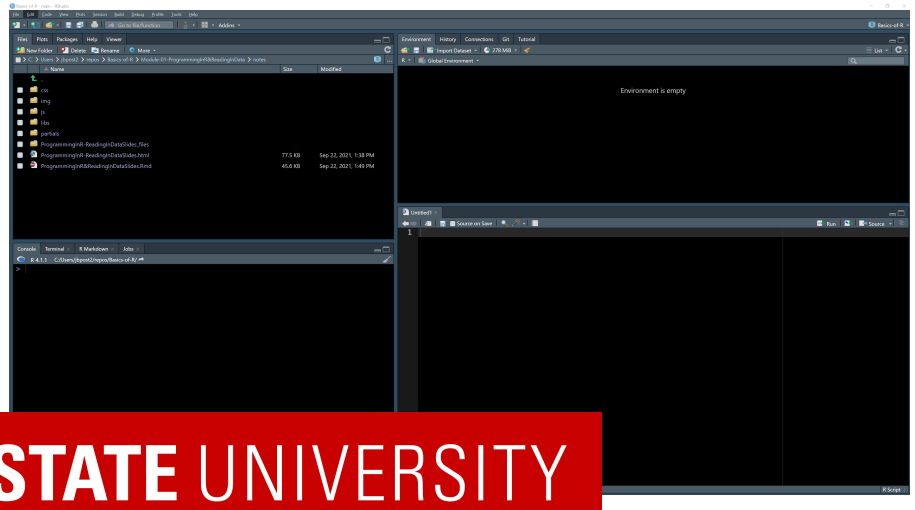
```
##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
```

- `data()` shows available built-in data sets

RStudio IDE

In RStudio, four main locations

- Console (& Terminal)
- Scripting and Viewing Window
- Files/Plots/Packages/Help
- Environment (& Connections/Git)



NC STATE UNIVERSITY

R Objects and Classes

How do we program in R? Using **objects** and **functions**

- R has strong **O**bject **O**riented **P**rogramming (OOP) tools

R Objects and Classes

How do we program in R? Using **objects** and **functions**

- R has strong **O**bject **O**riented **P**rogramming (OOP) tools
- Object: data structure with attributes (often a 'class')
- Method: procedures (often 'functions') act on object based on attributes

R Objects and Classes

Object: data structure with attributes (often a 'class')

Method: procedures (often 'functions') act on object based on attributes

- R functions like `plot()` act differently depending on object class

```
class(cars)
```

```
## [1] "data.frame"
```

```
class(exp)
```

```
## [1] "function"
```

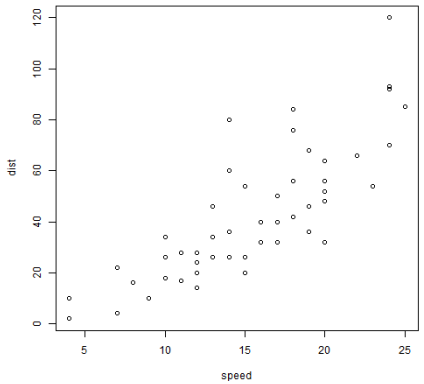
R Objects and Classes

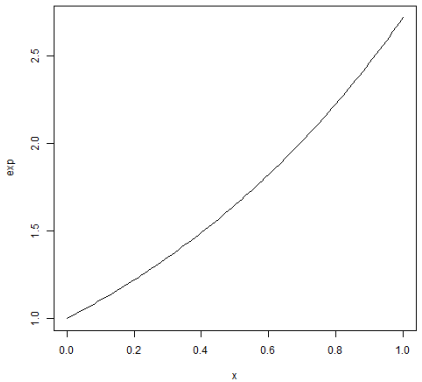
Object: data structure with attributes (often a 'class')

Method: procedures (often 'functions') act on object based on attributes

- R functions like `plot()` act differently depending on object class

```
plot(cars)
plot(exp)
```





R Objects and Classes

- Create an R object via `<-` (recommended) or `=`
  - allocates memory to object

```
vec <- c(1, 4, 10)
vec
```

```
## [1]  1  4 10
```

R Objects and Classes

- Create an R object via `<-` (recommended) or `=`
  - allocates memory to object

```
fit <- lm(dist ~ speed, data = cars)
fit
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Coefficients:
## (Intercept)        speed
##     -17.579        3.932
```

R Objects and Classes

- Function that creates the object determines the object's class

```
class(vec)
```

```
## [1] "numeric"
```

```
summary(vec)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0     2.5     4.0     5.0     7.0    10.0
```

R Objects and Classes

- Function that creates the object determines the object's class

```
class(fit)
```

```
## [1] "lm"
```

```
summary(fit)
```

```
##
## Call:
## lm(formula = dist ~ speed, data = cars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.069  -9.525  -2.272   9.215  43.201
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -17.5791     6.7584  -2.601   0.0123 *
## speed         3.9324     0.4155   9.464 1.49e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.38 on 48 degrees of freedom
##                           Adjusted R-squared:  0.6438
##                          DF,  p-value: 1.49e-12
```

Investigating Objects

Many functions to help understand an R Object

- `class()`
- describes the `class` attribute of an R object

```
class(cars)
```

```
## [1] "data.frame"
```

```
class(vec)
```

```
## [1] "numeric"
```

Investigating Objects

Many functions to help understand an R Object

- `typeof()`
- determines the (R internal) type or storage mode of any object

```
typeof(cars)
```

```
## [1] "list"
```

```
typeof(vec)
```

```
## [1] "double"
```

Investigating Objects

Many functions to help understand an R Object

- `str()`
- compactly displays the internal structure of an R object

```
str(cars)
```

```
## 'data.frame':    50 obs. of  2 variables:
##  $ speed: num  4 4 7 7 8 9 10 10 10 11 ...
##  $ dist : num  2 10 4 22 16 10 18 26 34 17 ...
```

```
str(vec)
```

```
##  num [1:3] 1 4 10
```

To R!

Quick example

- Customize the appearance of RStudio
- Check out the `help()` functionality
- Create object(s) and inspect them

Recap!

- RStudio provides a nice environment for coding
- R has functions that can be used to create objects
  - Create an R Object with `<-`
- Objects have attributes that determine how functions act!
  - `class()`, `typeof()`, and `str()` help understand your objects