About me

• BA in History, 2006; MS in Information Studies, 2011; MA in History in progress
• My background is in the humanities, not math, not science, not statistics, not computer science, not computer programming.

You do not have to be a statistician or programmer to learn and use R!
Session overview

• Session 1: Introduction to R and R Studio
• Session 2: Reading and Exploring Data
• Session 3: Analyzing and Visualizing Data
Poll Question

What programs do you currently use for your data cleanup, analysis & visualization?

Check all that apply. Type other answers into the Questions box.

- Excel
- Tableau
- Usage analysis tool provided by vendor (e.g. Intota Assessment, EBSCO Usage Consolidation)
- SPSS/SAS/other statistics program
- OpenRefine
This will be overwhelming!

Refer to the handout and stay positive!
Outline for Session 1: Nuts & Bolts

• About R
• About R Studio
• R Basics:
  • Writing & evaluating expressions
  • Assigning variables
  • Calling functions
  • Vectors
  • Data frames
• Troubleshooting errors
What is R?

• More of a programming language than a statistics program
• Originators described it as, "A language for data analysis and graphics"
• It can do
  • Data creation, importing & web scraping
  • Data cleaning & reshaping
  • Data visualization
  • Statistics & modeling
  • Text mining
  • Much more
Why R?

- Complex Data Manipulation
- Easier Automation
- Faster Computation
- Reads any kind of data
- Project organization
- Larger datasets
- Reproducibility & sharing

- Finding and fixing errors
- Free & open source
- Advanced statistics
- State-of-the-art graphics
- Runs on many platforms
- Active user community
- One-stop shop for
  - Data reading
  - Cleaning
  - Processing
  - Analyzing
  - Visualizing
  - Communicating

• Go to https://www.r-project.org/

• Click on CRAN (Comprehensive R Archive Network) under Download
  • scroll down to your country
  • select the download link corresponding to the city that is geographically closest to you
Download R

- Free and open source
  - GNU General Public License
  - Anyone can access the source code, build on it, improve it, and create packages & software to enhance user experience and functionality
Advantages of R

• Large user base in a variety of fields
  • Multiple academic disciplines: Statistics, Education, Genomics, Earth Sciences, Finance, Linguistics, Literature
  • Data science
  • Tech industry: Microsoft, Google, Facebook
  • Major companies: American Express, Ford, Uber
  • Journalism: *The New York Times*, *The Guardian*
  • Government: National Weather Service, Federal Drug Administration
• Getting help
  • Stack Overflow
  • R Project email lists

http://www.listendata.com/2016/12/companies-using-r.html
Drawbacks of R

- Objects must be stored in physical memory
- No Graphical User Interface: the command line can be intimidating!
  - If this stops you in your tracks, look at the R Commander software at www.rcommander.com, which provides a graphical user interface for R.

Roger Peng. Overview & History of R. https://www.youtube.com/watch?v=STihTnVSZnI
The new tool scale
Clean and merge messy data from the ILS

- Multiple repeated fields
  - e.g., order dates based on bib records

- Clean ISBNs & convert to 10 or 13 digit

- Detect data errors & anomalies

"0300017502;9780300017502;0300018509 (pbk.);9780300018509 (pbk.)"
R uses in libraries

Clean and merge messy data from vendors

• Normalizing names
• Removing/adding hyphens & other punctuation
• Creating custom subsets
• Merging data
  • Data fields often included in one dataset and not another
Recoding variables

<table>
<thead>
<tr>
<th>SUBJECT</th>
<th>subjRecode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic -- Study and teaching (Elementary) -- Mathematics</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Self perception -- Child psychology</td>
<td>Psychology</td>
</tr>
<tr>
<td>School music -- Instruction and study -- Music -- Handbell</td>
<td>Art</td>
</tr>
<tr>
<td>Geometry -- Study and teaching (Middle school)</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Readers -- 1050 -- Reading (Elementary) -- Reading (Primary)</td>
<td>Reading &amp; Language</td>
</tr>
<tr>
<td>Arithmetic -- Money -- Juvenile literature</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Nature study -- Ecology</td>
<td>NA</td>
</tr>
<tr>
<td>Arithmetic -- Remedial teaching -- Arithmetic -- Problem solving</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Mathematics -- Study and teaching (Primary) -- Mathematics</td>
<td>Mathematics</td>
</tr>
<tr>
<td>Language arts (Elementary) -- Creative writing (Elementary)</td>
<td>Reading &amp; Language</td>
</tr>
<tr>
<td>Poetics -- Study and teaching (Elementary)</td>
<td>NA</td>
</tr>
<tr>
<td>Language arts (Elementary) -- Reading (Elementary) -- Slip</td>
<td>Reading &amp; Language</td>
</tr>
<tr>
<td>Nature study -- Science -- Study and teaching (Elementary)</td>
<td>Science</td>
</tr>
</tbody>
</table>
R uses in libraries

Manipulating dates/times
R uses in libraries

Creating visualizations
R uses in libraries

• Analyzing collections
• Data reference
• Website/institutional repository analytics
• Library usage statistics
• Writing documents
**R Studio**

- A user interface for working with R

- You can use R without R Studio, but it's much more limiting

- R Studio makes it easier to
  - Import data, write and save scripts, install packages, work with objects

- Also free and open source
  - Download from rstudio.com
Panes in R Studio

- Text Editor or Script Pane
- Environment & History Pane
- Console Pane
- Navigation pane
Panes in R Studio

Console Pane – \textit{bottom left}

Type commands after the > prompt

```r
> y <- 5
> y + 5
[1] 10
> 
```
Panes in R Studio

Script Pane – *upper left*

Draft, run, and save scripts

```r
# This is the beginning of some very important code
# First we are going to assign the number 5 to y variable
y <- 5
# Then we are going to add 5 to it and assign that to the z variable
z <- y + 5
```
Panes in R Studio

Environment & History Pane – *upper right*

- Shows objects read into R
- Shows commands executed in the console
Panes in R Studio

Navigation pane – *bottom right*

- Navigate to your files
- View plots
- Load Packages
- See help files

![Example of an R Studio pane showing the `apply` function](image)
Panes in R Studio

Change pane size, layout, and appearance:

Go to Tools > Global Options
Writing expressions

• The **prompt** is the blinking cursor in the **console** pane.
• We type **expressions** into the prompt, and press the Enter key to **evaluate** them.

Command Prompt

```
> 2 + 2
[1] 4
```
Assign the number 5 to the symbol \( y \) using the assignment operator

\[
> \ y \ <- \ 5
\]

We say, "\( y \) gets 5."
Tips for assigning values

• Pick almost any name for your symbols...

ebook_usage <- read.csv("Fall ebook usage.csv", header = TRUE)

...except names of functions that already exist in R (e.g. avoid "sum" or "data")

> sum(sum)
Tips for assigning values

• R is **case sensitive**: a capital Y is not found

```r
> y <- 5
> Y
Error: object 'Y' not found
```

• Do not begin with numbers or symbols

```r
> 1z <- 4
Error: unexpected symbol in "1z"
> %z <- 4
Error: unexpected input in "%z <- 4"
```
Tips for assigning values

• No blank spaces or symbols other than underscore

```r
> my data <- 1:10
Error: unexpected symbol in "my data"
> myData <- 1:10
> my_data <- 1:10
```

• Be descriptive, but make your names short

```r
> this_variable_name_is_too_long <- 1:10
```
Evaluating expressions

• Press enter in the console to evaluate the expression and return the result

```
> y <- 5
> y
[1] 5
```

• The [1] means that the number 5 is the first element of this vector.
Evaluating expressions

- You can **assign** anything to any variable, and then perform operations on or with that variable

\[
\begin{align*}
> & \quad y \leftarrow 5 \\
> & \quad z \leftarrow c(5, 10, 15) \\
> & \quad y + 20 \\
& \quad [1] \quad 25 \\
> & \quad y^2 \\
& \quad [1] \quad 25 \\
> & \quad y + z \\
& \quad [1] \quad 10 \ 15 \ 20 \\
> & \quad \text{sum}(y, z) \\
& \quad [1] \quad 35
\end{align*}
\]
Calling a function

- R contains a number of **functions** you use to do something with your data
- **Call** a function on a variable by entering the function into the console followed by parentheses and the variables
- Functions can be **nested**

```r
> sum(3, 4)                          > sum(sqrt(9), 4)

> is.function(sum)
[1] TRUE
```
Calling a function

• Get help by typing a question mark followed by the function, and viewing the Navigation pane

> ?sum
> help(sum)
Calling a function

A function can take a specified number of arguments

```r
> sum(3, 4, 5, 6, 7)
[1] 25

> sum(3, 4, NA)
[1] NA

> sum(3, 4, NA, na.rm = TRUE)
[1] 7
```

**Sum of Vector Elements**

**Description**

`sum` returns the sum of all the values present in its arguments.

**Usage**

```r
sum(..., na.rm = FALSE)
```

**Arguments**

- `...` numeric or complex or logical vectors.
- `na.rm` logical. Should missing values (including NaN) be removed?
The `str()` function

`str()`: Compactly display information about an object

```r
> dogBreeds <- c("beagle", "pug", "chihuahua")
> str(dogBreeds)
chr [1:3] "beagle" "pug" "chihuahua"
```
The \texttt{c()} function

\texttt{c()}: combine values into a vector

\begin{verbatim}
> x <- c(1, 2, 3, 4, 5)
> str(x)
num [1:5]  1  2  3  4  5
\end{verbatim}
Objects & Vectors

• Everything you manipulate in R is called an **object**.
• Vectors are the most basic type of object
• A **vector** is a sequence of elements of the **same class**
Types of vectors

**numeric or integer**

```r
> my_integers <- 1:10
> str(my_integers)
int [1:10] 1 2 3 4 5 6 7 8 9 10
> class(my_integers)
[1] "integer"
```
Types of vectors

> my_characters <- c("Macbeth", "Dracula", "1984")

> str(my_characters)
chr [1:3] "Macbeth" "Dracula" "1984"

> class(my_characters)
[1] "character"
Types of vectors

**logical**

You can also use `T` and `F`

```r
> my_logical <- c(TRUE, FALSE, FALSE, TRUE)
> str(my_logical)
logi [1:4] TRUE FALSE FALSE TRUE

> is.logical(my_logical)
[1] TRUE
```
Types of vectors

**Factors**: Represent categorical data
- Unordered (nominal) or ordered (ordinal)

- Mostly useful for modeling functions in statistical analysis
- I usually coerce my factors to character strings when I read in data

```r
stringsAsFactors = F
```
Types of vectors

**factors**: represent categorical data
unordered (nominal) or ordered (ordinal)

```r

> table(my_nominal_factor)
my_nominal_factor
   book dissertation journal
      2          1          2
```
Types of vectors

**factors:** represent categorical data
ordered (ordinal)

```r
> my_ordinal_factor <- c("small", "medium", "large", "small", "large")

> my_ordinal_factor <- ordered(my_ordinal_factor,
  , levels = c("small", "medium", "large"))

> my_ordinal_factor
[1] small medium large small large
Levels: small < medium < large

> levels(my_ordinal_factor)
[1] "small" "medium" "large"
```
Subsetting a vector with brackets

```r
> scale <- c("do", "re", "mi", "fa", "so")
> scale[1]
[1] "do"
> scale[5]
[1] "so"

> scale[c(1, 5)]
> [1] "do" "so"

> scale[c(T, F, F, F, T)]
> [1] "do" "so"
```
• A **vector** can only contain objects of the same class...
• ...except for **lists**, which can contain multiple objects of different classes

```r
> my_list <- list(1:10,
                  , c("beagle", "pug", "chihuahua")
                  , TRUE)
> str(my_list)
List of 3
$ : int [1:10] 1 2 3 4 5 6 7 8 9 10
$ : chr [1:3] "beagle" "pug" "chihuahua"
$ : logi TRUE
```
Types of vectors

• If you mix different objects in one vector, R will **coerce** the vector to be a single class

```r
> y <- c(1:10, "a")
> str(y)
chr [1:11] "1" "2" "3" "4" "5" "6" "7" "8" "9" "10" "a"

> "1" + "2"
Error in "1" + "2" : non-numeric argument to binary operator
```
A way to deal with missing values

Use `is.na()` to test if a value is NA or not

Conversely, use `complete.cases()` to test if a value is not missing

```r
> vec <- c("value1", "value2", NA)

> is.na(vec)
[1] FALSE FALSE TRUE

> vec[is.na(vec)]
[1] NA

> vec[complete.cases(vec)]
[1] "value1" "value2"
```
Getting data into R

• Manually create it

• Import it from a file
  • Text file: TXT, CSV, TSV
  • Excel: XLSX
  • Statistics program: SPSS, MATLAB, Stata
  • Databases: MySQL
  • JSON, HTML, XML

• Gather it from the web
  • Connect to webpages, servers, or APIs
  • Webscraping
Data frames

• A grid of rows and columns
• Each column
  • is a vector of the same length
  • is a single data type
Data frames

```r
> ebooks <- data.frame("title" = c("Macbeth","Dracula","1984"),
                     "author" = c("Shakespeare","Stoker","Orwell"),
                     "checkouts" = c(25, 15, 18),
                     stringsAsFactors = F)
```
Data frames

You can **print** small data frames to the console...

```r
> print(ebooks)
   title     author checkouts
1  Macbeth  Shakespeare    25
2     Dracula    Stoker     15
3     1984      Orwell     18
```
Data frames

But for larger data frames it’s better to call the `view()` function

```r
> View(ebooks)
```

Or to click on the object in the Environment pane.
Can you edit data by hand as with spreadsheets?

Yes...But it’s best practice to script and document all modifications

> edit(ebooks)
Working in R Studio

• **Clear all objects** from the workspace: click the broom in the Environment/History (upper right) pane,

• **Clear specific objects** from the workspace

  > `rm(dogs)`

• **Clear console**: Press Ctrl + L (Command + L on Mac)

  or go to Edit > Clear Console
• While in the **console** (lower left pane), Press the up and down arrow keys on your keyboard to cycle through previously entered commands
To execute code from the **script pane**, highlight chunks of code and either

- Click the **Run** button
- Press Ctrl + R
You tried to run analysis on an object that has not been loaded into the environment. In other words, R can’t find what you’re looking for.

• Check for typos, capitalization, spelling, etc.
• Make sure the object has actually been loaded into R.
You tried use a function that doesn’t exist.

- Check for typos, capitalization, spelling, etc.
- Use `is.function()` to make sure it is actually a function.
- If the function is from a package that has not yet been loaded, load the package using `library()`
Troubleshooting errors

```
> sum(3, 4
   +
```

Hanging plus sign & blinking cursor: R is expecting something

- In this case, it’s waiting for me to close my parentheses. Could be waiting for closed quotes as well.
- Press the Escape key to interrupt the command, and add the required symbols.
You’ve entered a symbol in the wrong place.

- R stops evaluating the expression when it runs into the unexpected symbol—check there.
You made it!

Refer to the handouts for exercises and more in-depth explanations.
Session 2: Data Exploration

We will load packages, and read & explore data next session:

Wednesday, 5/23/2018

2:00 PM-3:00 PM (Eastern)
1:00 PM-2:00 PM (Central)
12:00 PM-1:00 PM (Mountain)
11:00 AM-12:00 PM (Pacific)

http://www.ala.org/alcts/confevents/upcoming/webinar/IntroToR