Introduction to R programming and data structures, Part I
Week 1, Lecture 01:

Richard E.W. Berl

Spring 2019

Contents

Introduction 1
Setting up a computing workflow 9
Basic concepts in R 10
Data types and classes 14
Data structures 15
Operations 16

Introduction

Who am I?

Mr. (almost Dr.!) Richard E.W. Berl

(but "Ricky" is fine)

I am an evolutionary social (data) scientist with a background in behavior and cultural change and a passion for conserving biocultural diversity and improving social good and environmental sustainability.

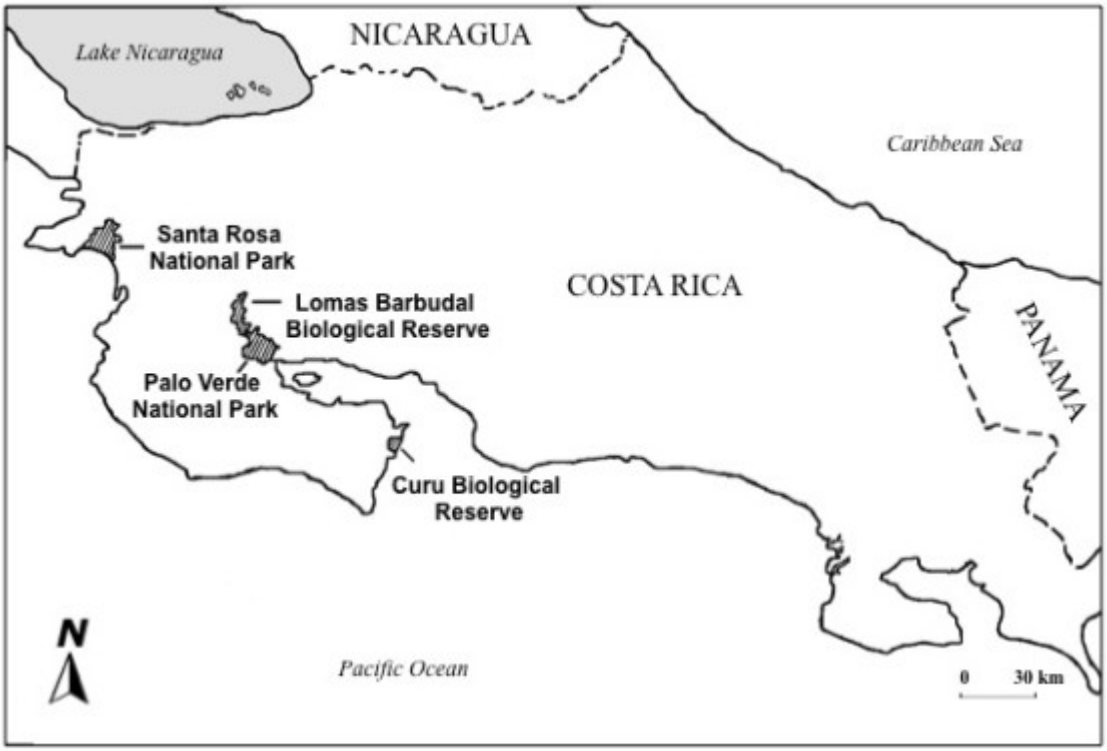
B.A. Biological Sciences & B.A. Anthropology from University of Delaware (2009)

Field Assistant, Lomas Barbudal Monkey Project (2009-2010)

- Lomas Barbudal Biological Reserve, Guanacaste, Costa Rica

Figure 1:

Figure 2:



– Source: UCLA Anthropology Lomas Barbudal Monkey Project

- Social learning and behavioral traditions
- Short documentary: *Family Trees* by Prehensile Productions

M.S. Zoology from Washington State University (2015)

- Social behavior and learning in captive gray wolves (*Canis lupus*) at Wolf Park in Battle Ground, IN

- Social behavior in wild gray wolves (*Canis lupus*) in Yellowstone National Park
- Overimitation in non-Western Central African societies

  - Source: Washington State University, Vancouver, Department of Anthropology

  - **Berl, R.E.W.** & Hewlett, B.S. 2015. Cultural variation in the use of overimitation by the Aka and Ngandu of the Congo Basin. PLOS ONE 10(3): e0120180. doi: 10.1371/journal.pone.0120180

    Number of irrelevant actions by group and demonstration condition.

- Cultural and genetic variation of the Chabu hunter-gatherers of Southwestern Ethiopia

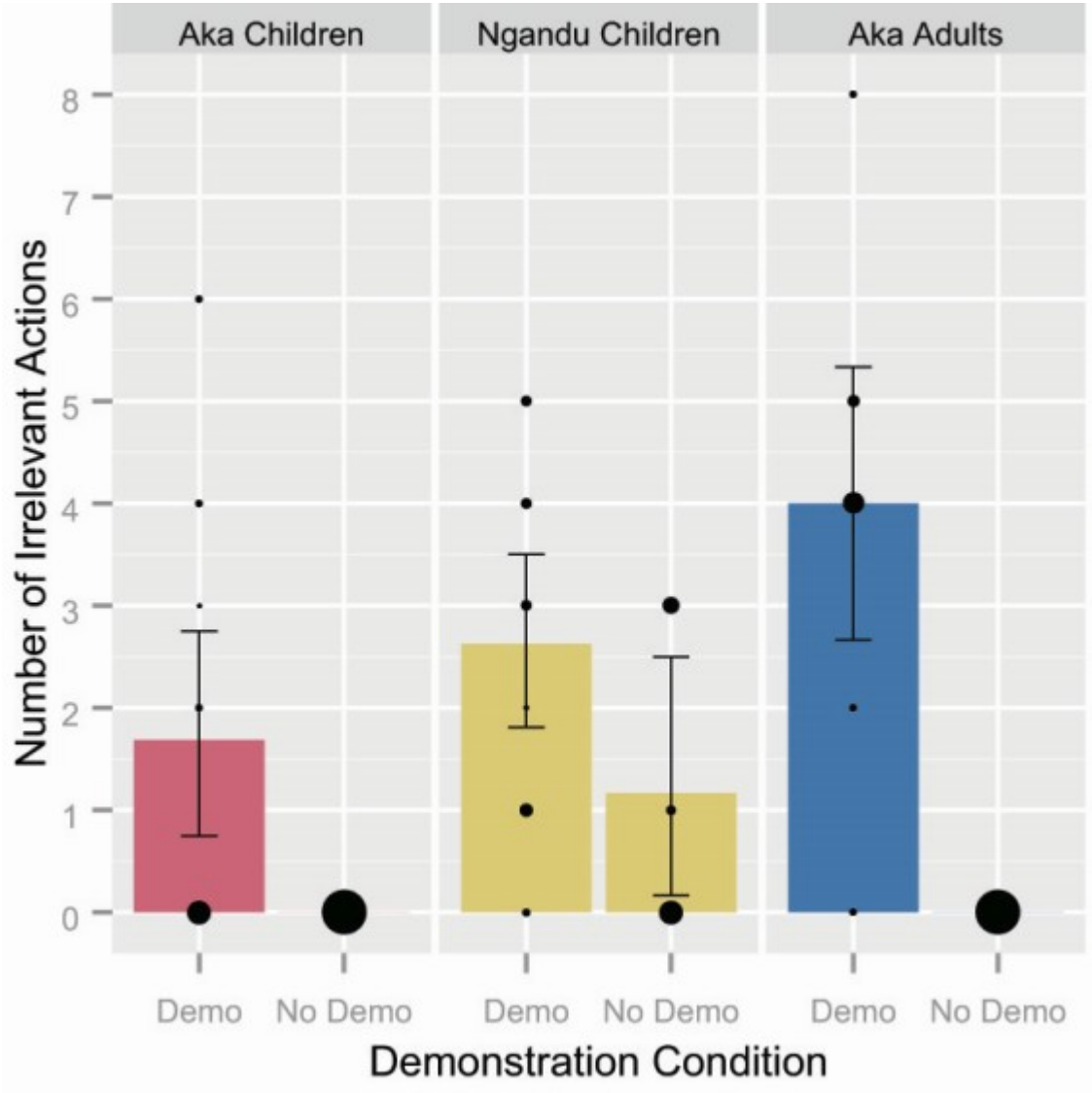


Figure 3:

Figure 4:

– Gopalan, S., **Berl, R. E. W.**, Belbin, G., Gignoux, C., Feldman, M. W., Hewlett, B. S., & Henn, B. M. (2019). Hunter-gatherer genomes reveal diverse demographic trajectories following the rise of farming in East Africa [preprint]. bioRxiv, 517730. Available: https://www.biorxiv.org/node/152746.abstract

Global ancestry proportions of northeast African individuals.

Effective migration surfaces depicted as contour lines over A) satellite imagery, B) elevation and water features, and C) the geographic distribution of major language families in Eastern Africa.

Ph.D. Human Dimensions of Natural Resources from Colorado State University (2019)

- Ph.D. Candidate in Human Dimensions of Natural Resources (defending on May 15th!)
- Graduate Certificate in Applied Statistics
- Influence of ***prestige*** in determining what people learn and from whom they choose to learn

Prestige domain item loadings from exploratory factor analysis of attitudinal data.

Determinants of prestige by level of social stratification across 16 societies.

Mean proportion of propositions recalled from artificial creation stories by type of content bias and by speaker prestige.

Color matrices of propositions recalled from artificial creation stories.

- Volunteer data scientist for Trees, Water & People

Random forest prediction of *Pinus ponderosa var. scopulorum* habitat suitability under present conditions on Pine Ridge Reservation and Trust Land.

Correlation matrix heatmap of climatic and soil variables.

Logistic regression of *Pinus ponderosa var. scopulorum* occurrence on burn area.

What we will cover in this course

- **See the Syllabus and Course Schedule**
- **Objectives (from Syllabus)**
  - Set up a convenient computing workflow
  - Write clean, thoroughly commented R code
  - Recognize different types of data, how they are measured, and how they are handled in R
  - Use the principle of 'tidy data' to effectively clean and format messy data sets

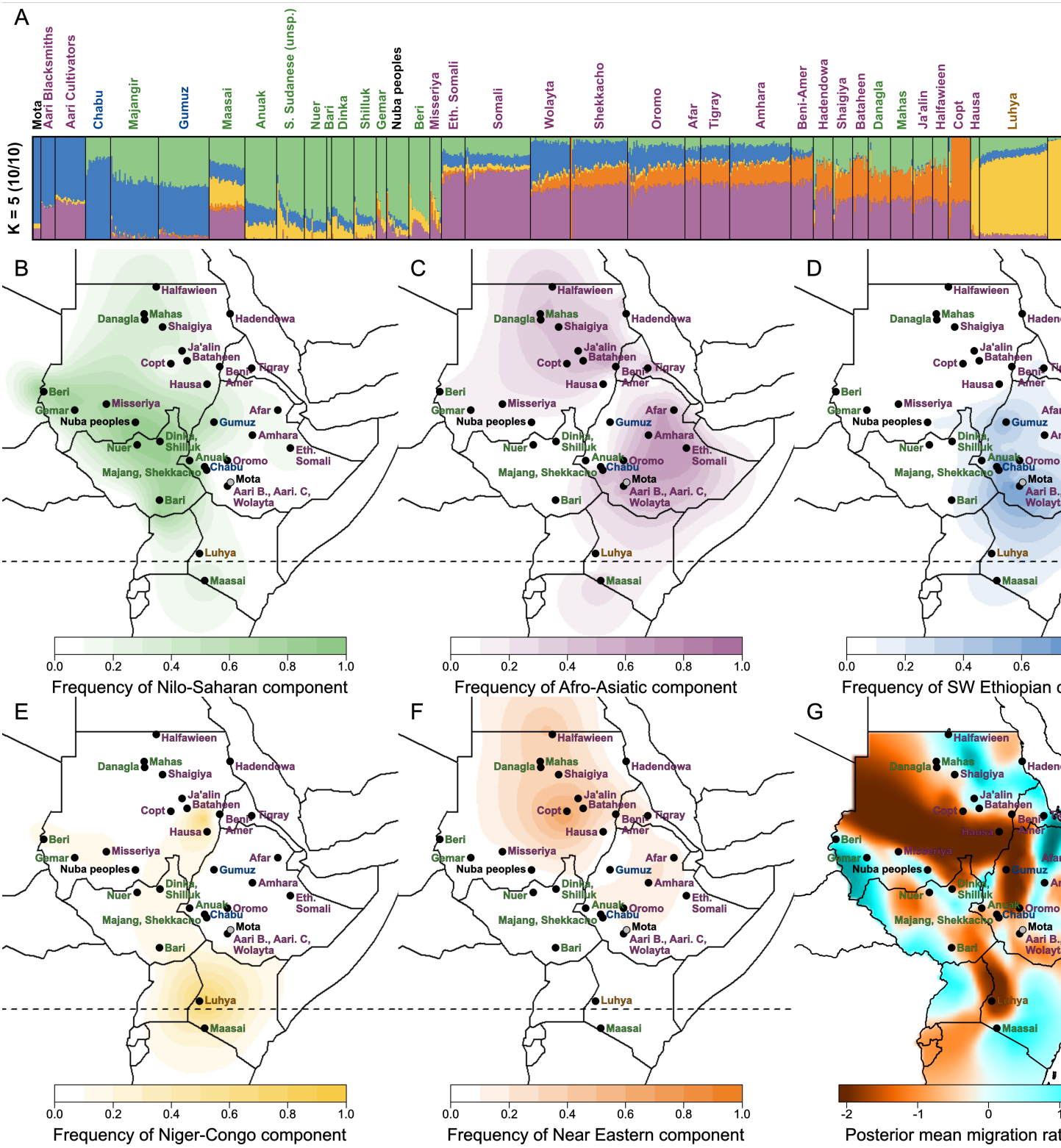

A
K = 5 (10/10)
Mota
Aari Blacksmiths
Aari Cultivators
Chabu
Majangir
Gumuz
Maasai
Anuak
S. Sudanese (unsp.)
Nuer
Bari
Dinka
Shilluk
Gemar
Nuba peoples
Beri
Misseriya
Eth. Somali
Somali
Wolayta
Shekkacho
Oromo
Afar
Tigray
Amhara
Beni-Amer
Hadendowa
Shaigiya
Bataheen
Danagla
Mahas
Ja'alin
Halfawieen
Copt
Hausa
Luhya
B
C
D
E
F
G
Frequency of Nilo-Saharan component
Frequency of Afro-Asiatic component
Frequency of SW Ethiopian c
Frequency of Niger-Congo component
Frequency of Near Eastern component
Posterior mean migration rat
0.0
0.2
0.4
0.6
0.8
1.0
-2
-1
0

Figure 5:

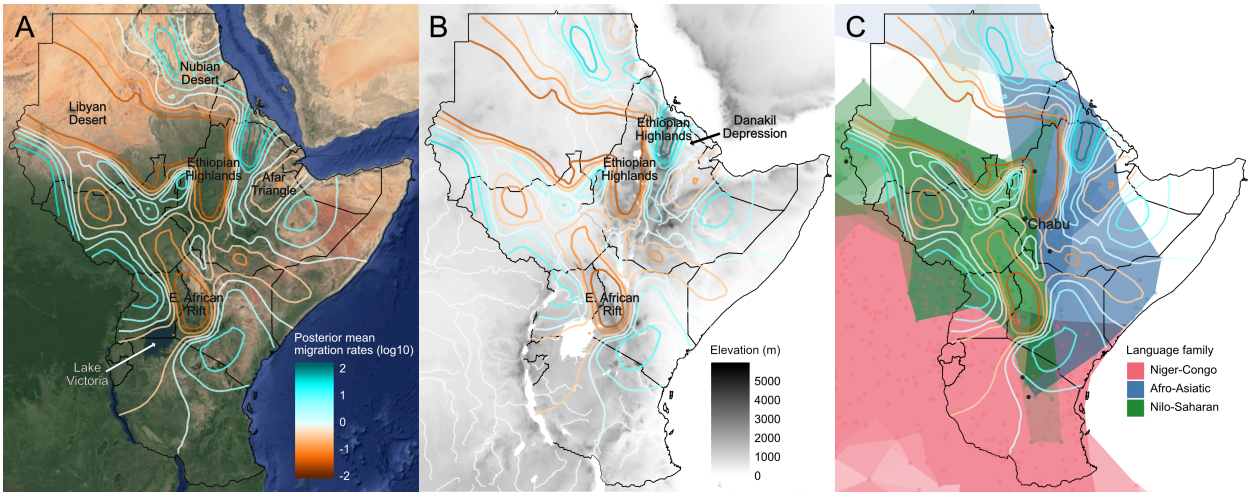


Figure 6:

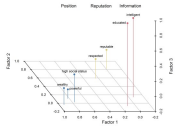


Figure 7:

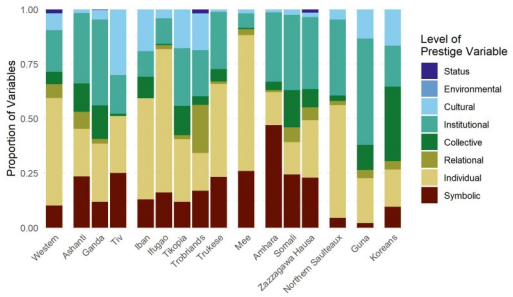


Figure 8:

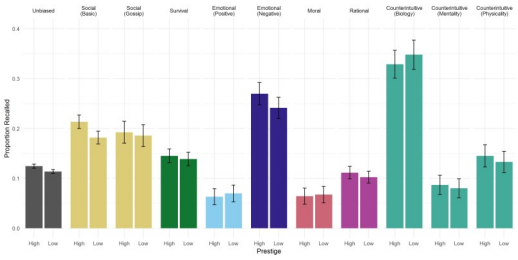


Figure 9:

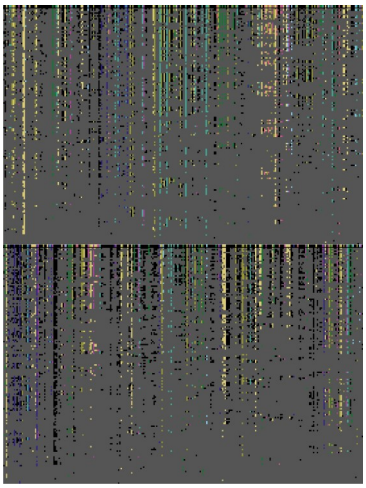


Figure 10:

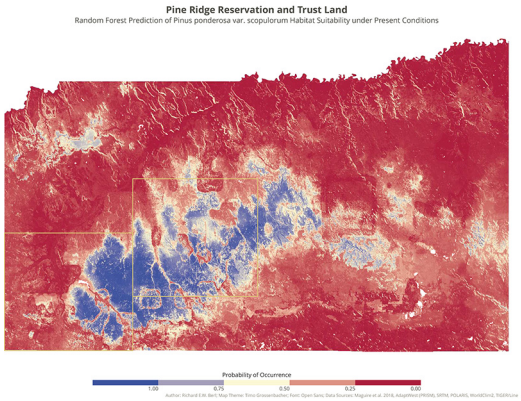


Figure 11:

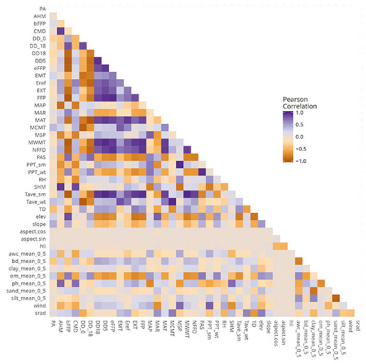


Figure 12:

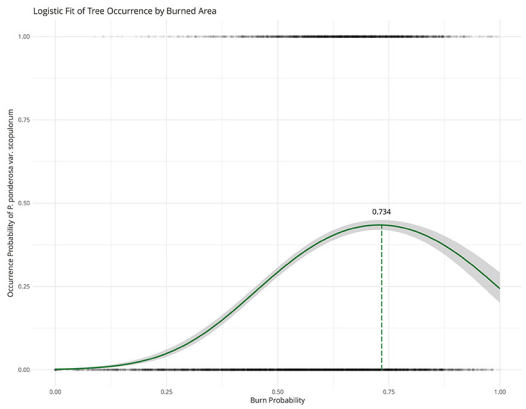


Figure 13:

- Creatively explore data sets with descriptive statistics and rough visualizations prior to confirmatory analyses
- Clearly communicate results by visualizing data simply and effectively and by telling a compelling story with data
- Conduct basic statistical tests and linear regression modeling
- Explore advanced topics in data analysis, including dimensionality reduction and structural equation modeling
- Utilize R for your own research by developing a research question, collecting and wrangling data, and conducting the appropriate analyses
- Support reproducible research by documenting and embedding analyses in a written report
- Use the skills you have learned to communicate your process and results to a general audience

What we will not cover

- R Markdown (*kind of*) and R Notebook
- LaTeX
- Version control
  - Git / GitHub
- Tibbles (`tibble` package) and piping (`magrittr` package)

Setting up a computing workflow

Good organization will save you time and frustration. Future you will thank present you. Trust me. You will have analyses that you'll come back to years later and wonder what in the world you were thinking. It also makes collaborative work a whole lot easier when your work is organized.

See the required reading by FitzJohn and their recommendations for organizing a project. Do it for this class and use it to help organize your own research projects, as well. The earlier, the better.

Create a folder structure for this course. I'd recommend something like the following:

```
nr592/
├── data/
├── docs/
├── figs/
├── output/
└── R/
    ├── assignment1.R
    └── lecture01.R
```

You may want to create another folder, like `lectures\`, to keep my R Markdown lecture files in (the `lecture01.R` in the structure above refers to your own notes). Or you could put them in `docs\`.

All of the above are just suggestions. Do what makes sense to you and stick with it.

Now we're going to create an R project file for the course. You should put it in your main class folder, like this:

```
nr592/
├── data/
├── docs/
├── figs/
├── output/
├── R/
└── nr592.Rproj
```

Use the button in the top right that says "Project: (None)" and select "New Project..." Choose "Existing Directory" if you've already made the class folder, as above. Navigate to the class folder and hit "Create Project."

You're done! Any time you need to work on something for this class, you can open this project and any scripts you had open last time will open up for you.

Importantly, your working directory (the directory where R will look for any files you're loading or saving) is set to your class directory while you are working on scripts in your R project.

You can check your working directory at any time by running:

```
getwd()
```

```
## [1] "S:/MEGAsync/CSU/Courses/NR 592 (R Seminar)/Site/lectures"
```

(This is my current working directory.)

And if you need to change it, you can do so using `setwd()` with the full folder path inside quotes, inside the parentheses, like so:

```
setwd("C:/My Research/My Big Project/")
```

(Note the direction of the slashes.)

Basic concepts in R

DON'T BE AFRAID TO FAIL!

Type something and run it! It doesn't matter if you get an error; you won't break anything.

Run current line/selection of code:

- Ctrl+Enter (Windows)
- Command+Enter (Mac)

Source: RStudio Keyboard Shortcuts

Objects

Variables

Assignment

```
a = 1
b = 2
c = 42
```

```
a
```

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

```
b
```

```
## [1] 2
```

```
c
```

```
## [1] 42
```

Assignment can also be done using the `<-` operator (i.e. `c <- 42`). I like using `=` because it uses fewer keystrokes and is more similar to other programming languages, like Python. You can use either, just pick one and stick with it.

If you use `=`, keep in mind that `=` is used for assignment and `==` is used for logical comparisons, i.e. "Does `a == 7`? `FALSE`".

If you use `<-`, the direction of the arrow matters: `c <- 42` is the same as `42 -> c`, but is different from `c -> 42`. You can also assign a value to multiple variables at once with this operator, i.e. `g <- h <- i <- 6`.

Operations

```
a + b
```

```
## [1] 3
```

```
a^2 + b^2
```

```
## [1] 5
```

```
c / (a + b)
```

```
## [1] 14
```

Spacing doesn't matter.

```
c / (a                                                                  +b )
```

```
## [1] 14
```

You can even hit 'Enter'/'Return' in a script (see below) and carry on to the next line.

```
(a - c) +
  (a * b) +
  (c / b)
```

```
## [1] -18
```

Or leave empty lines between commands (though you wouldn't want to do this in the middle of a command–it would be confusing and *bad stuff* could happen).

```
a + b + c - b^2
```

```
c * 2
```

(This code block wasn't evaluated so that you could actually see the spacing, but it works! Try it.)

Use spacing to your advantage to make your code more easily readable. If you have a line of code that goes over 80 characters (the thin line on the right side of your script pane), insert returns after operators (+, -, etc.) for clean breaks in your code. R will auto-indent the next line for you.

Reassignment

```
a
```

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

```
b
```

```
## [1] 2
```

```
a = a + b
a
```

```
## [1] 3
```

```
a = a + b
a
```

```
## [1] 5
```

```
a = a + b
a
```

```
## [1] 7
```

What is the value of `b`?

```
b
```

```
## [1] 2
```

Why didn't `b` change?

We never reassigned it (`b = ...`), only the value of `a`.

Scripts

Always work in scripts!

Open a new R script:

- Ctrl+Shift+N (Windows)
- Command+Shift+N (Mac)

Commenting

```
# "This line is commented out."
"This line is not commented out."
```

```
## [1] "This line is not commented out."
```

Comment/uncomment line:

- Ctrl+Shift+C (Windows)
- Command+Shift+C (Mac)

An example of well-commented code:

```
# Load iris data
data(iris)

# View structure of iris data
str(iris)
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# Subset iris data to Species versicolor
irisVe = subset(x=iris, subset=Species == "versicolor")

# Find correlation between sepal length and petal length in versicolor
cor(x=irisVe$Sepal.Length, y=irisVe$Petal.Length)  # Result: 0.754049
```

```
## [1] 0.754049
```

You can read about the iris data set by entering `?iris` in the console. Notice where R. A. Fisher's paper was published. We'll address this, and the dark side of the origins of statistics, in Week 4.

Packages

Packages are collections of functions and data sets useful for conducting different types of analyses.

You can get a list of packages that come installed with R using the command (more detail available by removing the `rownames()` wrapper):

```
rownames(installed.packages(priority="high"))
```

```
##  [1] "nlme"       "base"       "boot"       "class"      "cluster"
##  [6] "codetools"  "compiler"   "datasets"   "foreign"    "graphics"
## [11] "grDevices"  "grid"       "KernSmooth" "lattice"    "MASS"
## [16] "Matrix"     "methods"    "mgcv"       "nlme"       "nnet"
## [21] "parallel"   "rpart"      "spatial"    "splines"    "stats"
## [26] "stats4"     "survival"   "tcltk"      "tools"      "utils"
```

Additional packages can be installed using the `install.packages()` function, for example:

```
install.packages("ggplot2")
install.packages(c("dplyr","tidyr"))
```

Notice that if you're installing multiple packages at once, the list of packages **must** be given as a vector.

Load packages using the `library()` function:

```
library(ggplot2)
```

You can only load packages one at a time. Sometimes load order matters, because if two packages have functions with the same name, the last one loaded is the one R will assume you are referring to. To specify which one you want, you can put the package name before your function call, like so: `dplyr::arrange()`.

`install.packages()` has to have the package name in quotes. For `library()`, quotes are optional.

Lists of packages useful for specific tasks, such as machine learning or Bayesian inference, are available on the Task Views page of CRAN ("Comprehensive R Archive Network"), which is an official site run by the developers of R.

Data types and classes

Nominal (or Categorical)

"A nominal scale variable (also referred to as a categorical variable) is one in which there is no particular relationship between the different possibilities: for these kinds of variables it doesn't make any sense to say that one of them is"bigger" or "better" than any other one, and it absolutely doesn't make any sense to average them... In short, nominal scale variables are those for which the only thing you can say about the different possibilities is that they are different. That's it."

Source: Navarro, Section 2.2

In R these are:

- Character
- Factor

Ordinal (or Ranked)

"Ordinal scale variables have a bit more structure than nominal scale variables, but not by a lot. An ordinal scale variable is one in which there is a natural, meaningful way to order the different possibilities, but you can't do anything else."

Source: Navarro, Section 2.2

In R these are:

- Ordered factor

Measurement

Continuous

"A continuous variable is one in which, for any two values that you can think of, it's always logically possible to have another value in between."

Source: Navarro, Section 2.2

In R these are:

- Numeric

Discrete

"A discrete variable is, in effect, a variable that isn't continuous. For a discrete variable, it's sometimes the case that there's nothing in the middle."

Source: Navarro, Section 2.2

In R these are:

- Integer
- Logical (*sometimes*)

Other types

See the following for a list of all basic classes included in R:

```
?methods::`character-class`
```

Dates and Times

In R these are:

- Date
- POSIXct
- POSIXlt

For example, see:

```
Sys.Date()
```

```
## [1] "2019-03-26"
```

```
str(Sys.Date())
```

```
##  Date[1:1], format: "2019-03-26"
```

```
Sys.time()
```

```
## [1] "2019-03-26 02:40:59 MDT"
```

```
str(Sys.time())
```

```
##  POSIXct[1:1], format: "2019-03-26 02:40:59"
```

These are **awful** to deal with in R. Avoid doing analyses on dates and times whenever possible. If you must, look into the `lubridate` package to make things a bit easier.

Coercion

Data structures

Vectors

Can only have one type

Matrices

Data Frames

Lists

Functions

```
a
```

```
## [1] 7
```

```
b
```

```
## [1] 2
```

```
c
```

```
## [1] 42
```

```
mean(x=c(a, b, c))
```

```
## [1] 17
```

Let's make our own function to add the variable `a` to the variable `b`.

```
a_plus_b = function(a, b){
  aPlusB = a + b
  return(aPlusB)
}
```

```
a_plus_b(a=1, b=5)
```

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

What if we just use `aPlusB` instead of the whole function?

```
aPlusB(a=1, b=5)
```

```
## Error in aPlusB(a = 1, b = 5): could not find function "aPlusB"
```

Why doesn't this work?

`aPlusB` is a variable, not a function. You can't pass other variables to it.

Let's see what value is stored for the `aPlusB` variable.

```
aPlusB
```

```
## Error in eval(expr, envir, enclos): object 'aPlusB' not found
```

Why doesn't this work?

`aPlusB` only exists inside the `a_plus_b` function. It doesn't have a value assigned to it in the "global environment" that we're working in. (Look in the "Environment" tab in RStudio.) `aPlusB` is called an "internal variable" because it only assigned inside the function when it's called, and is removed as soon as the function is finished running.

Operations

Common Functions

str() names() class() dim() length() nrow() ncol() rownames() colnames() gc() ? View() rm()

Subsetting

Conditionals

Iteration

Loops

Apply

(pdf / Rmd)