# Section 0

*Foundations of Statistical Inference (PLSC 503)*

Welcome to PLSC 503! The goal of this section is to get you all up an running in `R`. You will need to bring a laptop to all sections with `R` and RStudio installed. This is so that we can work through examples together. If this presents a problem for anyone please let us know asap!

Hikaru Yamagishi

- Email: hikaru.yamagishi@yale.edu
- Office Hours: Wednesday 10am - 12pm in Rosenkranz Hall (Room 204)

Kyle Peyton

- Email: kyle.peyton@yale.edu
- Office Hours: Friday 9am-11am in Institution for Social and Policy Studies (Room C321)

## Step 1: R

Please go here https://cran.r-project.org/ to download the latest version of `R` and install it on your machine.

## Step 2: RStudio

Please go here https://www.rstudio.com/products/rstudio/download/ and download the **free** desktop version on RStudio on your machine.

## Step 3: Latex

If you want to compile this .Rmd document as a PDF then you need to get Latex installed on your machine. It will still compile as an html or word doc just fine without it. However, it is recommended that you get Latex up and running on your machine and generate PDFs for sharing your work, not word documents. You can still work in Markdown and RStudio, Latex is just needed for some behind the scenese stuff.

First, you need to install a Latex distribution from https://www.latex-project.org/. Next, follow the instructions below for your operating system. If you have problems with the instructions below, please reach out to us for help.

- **Windows**: download the "MiKTex" distribution available at https://www.latex-project.org/get/.
- **Mac**: download the "MacTeX" distribution available at https://www.latex-project.org/get/.

## Step 4: Workflow

We highly recommend using Markdown for all your homeworks. We promise it will make your life easier in the long run. It can generate PDFs and Word documents. You can also write in Latex syntax. But the Markdown syntax if very straightforward. The integration with RStudio is seamless. There are many many introductions to Markdown online. Here are two useful reference documents:

- https://www.rstudio.com/wp-content/uploads/2016/03/rmarkdown-cheatsheet-2.0.pdf
- https://www.rstudio.com/wp-content/uploads/2015/03/rmarkdown-reference.pdf

This might seem intimidating, but getting started is super easy. Don't look at the references until you need them. Just dig in:

- Open RStudio
- File > New File > R Markdown ...
- Now you have created your first Markdown (.Rmd) file! Here is a concise and excellent introductory tutorial: http://r4ds.had.co.nz/r-markdown.html

This document was created in Markdown using some Latex syntax to create math. So all the code chunks you see in the next section are things that you can also do very easily. Have a look at the .Rmd file as well so that you can see the code that generates this PDF. It could also serve as a template for your homeworks if you want. Some of us use Latex syntax in Markdown as a matter of habit, but .Rmd files are very flexible, so you can use a combination of Markdown and Latex syntax.

Finally, be kind to your future self. Write good code. This makes your life easier, and the lives of people who have to read your code. And it also signals to other people who might read your code (while replicating your work perhaps) that you are careful and well organized. There are many style guides out there. Have a look at this one: http://adv-r.had.co.nz/Style.html. The Google Style guide is also useful: https://google.github.io/styleguide/Rguide.xml.

If you haven't seen `R` before, this might look scary. But learning by example is the very best way to get started. This tweet https://twitter.com/kierisi/status/898534740051062785?s=03 is a good description of the basic workflow in `R`:

1. Install R
2. Install RStudio
3. Google "How do I [THING I WANT TO DO] in R?"
4. Repeat 3 as necessary.

## Useful resources

There are many good introductions to `R` on the internet. Some of my favorite books (in order of difficulty) are,

1. R for Data Science (https://r4ds.had.co.nz/). A problem oriented introduction to data analysis + R via Hadley Wickham' universe of awesome packages.
2. Introduction to Probability and Statistics Using R (https://cran.r-project.org/web/packages/IPSUR/vignettes/IPSUR.pdf). This is a very comprehensive (free) introduction. See Chapter 2 for the basics.
3. The Art of R Programming (https://www.nostarch.com/artofr.htm). I think this is available through Yale Library as an e-book. It's a very useful reference to have.
4. The R Infero (http://www.burns-stat.com/pages/Tutor/R/_inferno.pdf). Author's description: "If you are using R and you think you're in hell, this is a map for you." This is a very quirky book, and I found it useful (and funny) when I first started learning.
5. Advanced R (http://adv-r.had.co.nz/). This is a free resource from Hadley Wickham. It's a very good reference for advanced topics.

There are also several good websites to check out for help. These are my favorites,

- https://stackoverflow.com/questions/tagged/r
- https://www.r-bloggers.com/

## Course Repository

We will often add to or revise section notes after we have met in order to incorporate feedback and answer questions. Visit the PLSC 503 Github repository for the most up to date section materials: https://github.com/kylepeyton/PLSC503.

# Practice Activities

### Activity 1

Write a sentence about what you hope to gain from this course that uses *italics* and **bold**.

I **hope** to gain more experience teaching statistics *things*.

### Activity 2

Make an unordered list of things using bullet points.

- This
- Is
- a
- list
- of things.

### Activity 3

Make an ordered list of things using numbers.

1. This
2. is
3. number
4. four?

### Activity 4

Write the following equation in math mode:

$y = \alpha + \beta X + \gamma Z$

### Activity 5

Load the csv from http://hdl.handle.net/10079/dfn2zdg using two different methods.

```
# Method 1. Specify file location. Recommended.
path <- "/Users/kylepeyton/downloads/"
toy_df <- read.csv(paste0(path, "GerberGreenBook_Chapter2_Table_2_1.csv"))

# Method 2. Set your working directory to where the file lives. Not recommended.
# See: https://twitter.com/hadleywickham/status/940021008764846080
setwd("/Users/kylepeyton/downloads/")
toy_df <- read.csv("GerberGreenBook_Chapter2_Table_2_1.csv")

# Method 3. Direct from URL. Highly recommended.
toy_df <- read.csv("http://hdl.handle.net/10079/dfn2zdg")
```

### Activity 6-7

Install the haven package. Load the Stata .dta from http://hdl.handle.net/10079/1g1jx43 using your favorite method.

```
# First need to install this package if you don't have it already:
# install.packages("haven")

# Load the package and read in a Stata dataset using Method 3 from above.
# Note you can use ?read_dta() to learn more about this function.
library(haven)
toy_df <- read_dta("http://hdl.handle.net/10079/1g1jx43")
```

### Activity 8

There are many datasets in base `R`. Let's load the chickens dataset and do some basic stuff.

```
# Load dataset
data("chickwts")

# What is this thing?
?chickwts
```

In `R` a dataset is a "data frame". `R` also understands matrices; but data frames have the useful property that they can be a combination of character strings and numerics.

```
# Three types of domestic cats
cats <- c("Ragdoll", "Siberian", "Main Coon")

# Ranking of their adult sizes
rank <- c(2, 3, 1)

# Make a dataframe
cat_df <- data.frame(cats = cats, rank = rank)

# Print it out
cat_df
```

```
##        cats rank
## 1   Ragdoll    2
## 2  Siberian    3
## 3 Main Coon    1
```

The `class()` function tells you what class (e.g. data.frame, matrix) an object belongs to:

```
class(chickwts)
```

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

```
class(cat_df)
```

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

The function `head()` is useful for having a peek at the top of a data frame. If you want to look at the entire data frame in spreadsheet format, you can used the `View()` function. This is not recommended for "large" data frame. Even something with ~40k rows and 100 columns will give you trouble on most machines.

```r
# Take a look at the first 6 rows. You can pass the argument n = 10 to print
# the first 10 rows. Check out ?head() to see what arguments the function takes.
head(chickwts)
```

```
##   weight      feed
## 1    179 horsebean
## 2    160 horsebean
## 3    136 horsebean
## 4    227 horsebean
## 5    217 horsebean
## 6    168 horsebean
```

`nrow()` is another basic function that comes in handy. As the name suggests, it tells you how many rows are in a data frame.

```r
# How many observations are in here?
nrow(chickwts)
```

```
## [1] 71
```

## Activity 9

Compute some summary statistics. The `summary()` function is useful for datasets that do not have too many columns.

```r
# Compute summary statistics for each column
summary(chickwts)
```

```
##      weight             feed
##  Min.   :108.0   casein   :12
##  1st Qu.:204.5   horsebean:10
##  Median :258.0   linseed  :12
##  Mean   :261.3   meatmeal :11
##  3rd Qu.:323.5   soybean  :14
##  Max.   :423.0   sunflower:12
```

You can also access variables from within the dataset using the money sign:

```r
# Calculate the variance of chicken weights
var(chickwts$weight)
```

```
## [1] 6095.503
```

```r
# Calculate the standard deviation
sd(chickwts$weight)
```

```
## [1] 78.0737
```

```r
sqrt(var(chickwts$weight))
```

```
## [1] 78.0737
```

The feed type is stored as a factor. This is an important `R` object. It's like a character string, but it has an ordering to it. By default the ordering is alphabetical.

```r
# Feed type is a factor:
class(chickwts$feed)
```

```
## [1] "factor"
```

Let's have a closer look at this variable:

```
# Tabulate the feed types
table(chickwts$feed)
```

```
##
##    casein horsebean   linseed  meatmeal   soybean sunflower
##        12        10        12        11        14        12
```

```
# It has six levels
levels(chickwts$feed)
```

```
## [1] "casein"    "horsebean" "linseed"   "meatmeal"  "soybean"   "sunflower"
```

The six levels tell us how the factor is ordered. We can convert factors to numerics or characters. We cannot convert a character to a numeric, however. This is because it does not have a natural ordering, unless you tell R to give it one.

```
# Convert to character, and store this new variable inside the dataset as
# a new column
chickwts$feed_char <- as.character(chickwts$feed)
class(chickwts$feed_char)
```

```
## [1] "character"
```

```
# This will produce an error because factors don't have an implicit ordering:
as.numeric(chickwts$feed_char)
```

```
## Warning: NAs introduced by coercion
```

```
##  [1] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [24] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [47] NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA NA
## [70] NA NA
```

Note the ordering when we convert the factor to a numeric, "c" comes before "h" in the alphabet, so "casein" gets an 1 and "horsebean" gets a 2, etc.

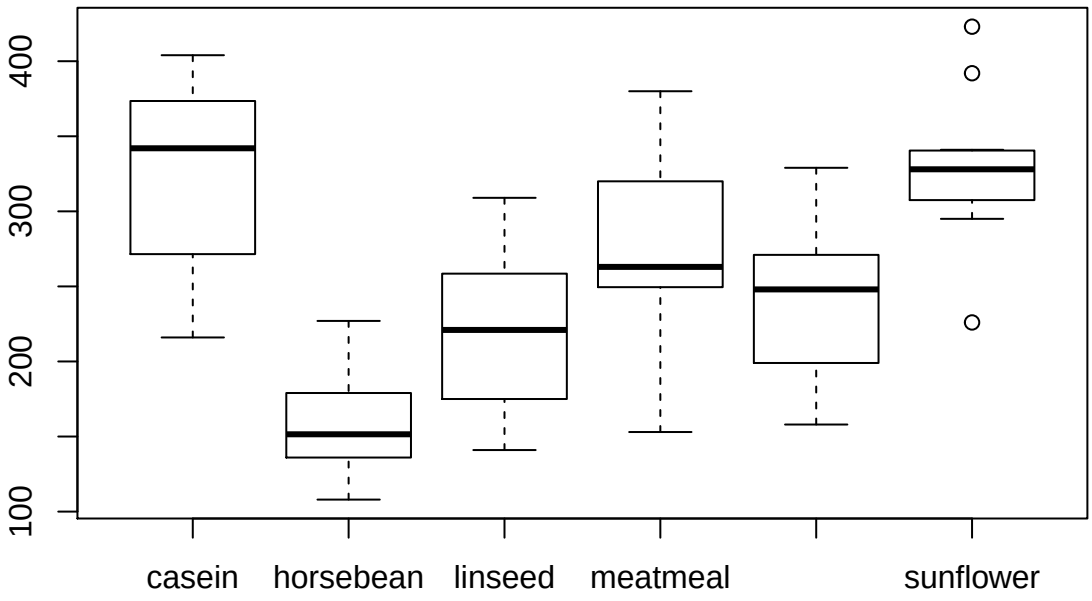
```
# Convert factor to numeric, and cross-tabulate the two vectors
table(as.numeric(chickwts$feed), chickwts$feed_char)
```

```
##
##     casein horsebean linseed meatmeal soybean sunflower
##   1     12         0       0        0       0         0
##   2      0        10       0        0       0         0
##   3      0         0      12        0       0         0
##   4      0         0       0       11       0         0
##   5      0         0       0        0      14         0
##   6      0         0       0        0       0        12
```

### Activity 10

Make a plot in base R. Let's make a boxplot using the "formula" notation. This takes a numeric outcome on the left hand side (displayed on vertical axis in the plot) and a factor on the right hand side (displayed on horizontal axis in the plot)

```
# Make a boxplot; see ?boxplot for more info
boxplot(weight ~ feed, data = chickwts)
```


100
200
300
400
casein
horsebean
linseed
meatmeal
sunflower