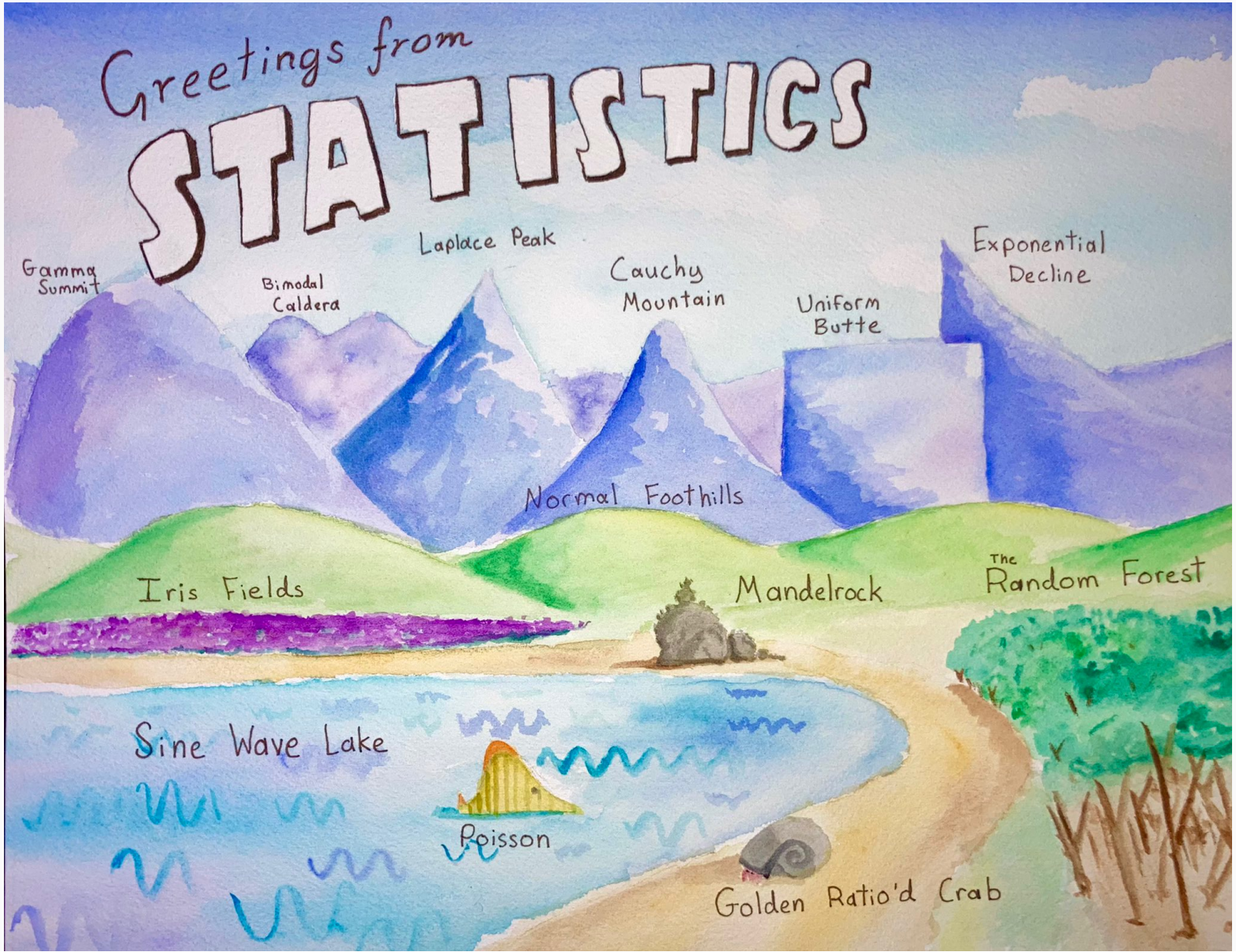
Introduction

EPSY 630 - Statistics II

Jason Bryer, Ph.D.

Spring 2021


Greetings from
STATISTICS
Gamma Summit
Bimodal Caldera
Laplace Peak
Cauchy Mountain
Uniform Butte
Exponential Decline
Normal Foothills
Iris Fields
Mandelrock
The Random Forest
Sine Wave Lake
Poisson
Golden Ratio'd Crab

@skyetetra

Agenda

- About your instructor
- Syllabus
- Class meetups
- Course Schedule
- Assignments (how you will be graded)
  - Participation
  - Homework
  - Labs
  - Data Project
  - Final exam
- Software
  - Getting started with R
  - The `DATA606` R Package
  - Using R Markdown

Introduction

A little about me:

- Assistant Professor at CUNY in Data Science and Information Systems
- Principal Investigator for a Department of Education Grant (part of their FIPSE First in the World program) to develop a Diagnostic Assessment and Achievement of College Skills (www.DAACS.net)
- Authored over a dozen R packages including:
  - likert
  - sqlutils
  - timeline
- Specialize in propensity score methods. Three new methods/R packages developed include:
  - multilevelPSA
  - TriMatch
  - PSAboot
- Developer of a data dashboard for the NYS Office of Special Education and TAP for Data at Cornell University: https://data.osepartnership.org

Also a Father...

Runner...

And photographer.

Your turn

Your name?

What program are you in and level (Master's or Doctorate)?

Something we wouldn't otherwide know about you?

Syllabus

Syllabus and course materials are here: https://epsy630.bryer.org

We will use Blackboard primary for submitting assignments only.

The site is built using the Blogdown R package and hosted on Github. Each page of the site has a "Improve this page" link at the bottom right, use that to start a pull request on Github.

Class Meetings

We will have meetups on Tuesdays evenings at 4:30pm.

Class meetings will be recorded and made available the next day on the course website.

One Minute Papers - Complete the one minute paper after each Meetup (whether you watch live or watch the recordings). It should take approximately one to two minutes to complete. This allows me to 1) verify you have attended/watch the meetup and 2) get feedback about what you learned and what you may still be unclear.
Link: https://forms.gle/gY9SeBCPggHEtZYw6

Please note: *Students who participate in this class with their camera on or use a profile image are agreeing to have their video or image recorded solely for the purpose of creating a record for students enrolled in the class to refer to, including those enrolled students who are unable to attend live. If you are unwilling to consent to have your profile or video image recorded, be sure to keep your camera off and do not use a profile image. Likewise, students who un-mute during class and participate orally are agreeing to have their voices recorded. If you are not willing to consent to have your voice recorded during class, you will need to keep your mute button activated and communicate exclusively using the "chat" feature, which allows students to type questions and comments live.*

Schedule*

Date	Topic
Tuesday, February 02, 2021	Intro to the Course / Intro to R and RStudio
Tuesday, February 09, 2021	Descriptive Statistics / Data Visualizaiton
Tuesday, February 16, 2021	Central Limit Theorem
Tuesday, February 23, 2021	Null hypotheis testing / confidence intervals /boostrapping
Tuesday, March 02, 2021	Linear Regression
Tuesday, March 09, 2021	Multiple Regression
Tuesday, March 16, 2021	Maximum Likelihood Estimation / Logistic Regression
Tuesday, March 23, 2021	ANOVA - Chi-Squared Tests
Tuesday, March 30, 2021	TBD
Tuesday, April 06, 2021	NO CLASS
Tuesday, April 13, 2021	TBD
Tuesday, April 20, 2021	Propensity Score Analysis
Tuesday, April 27, 2021	Bayesian Analysis
Tuesday, May 04, 2021	Project Presenations
Tuesday, May 11, 2021	Project Presenations

*Tentative. Subject to change.

Textbooks

Diez, D.M., Barr, C.D., & Çetinkaya-Rundel, M. (2019). *OpenIntro Statistics (4th Ed).*

This will be our primary textbook for most of the semesters. Our goal is to cover all the chapters.

Navarro, D. (2018, version 0.6). *Learning Statistics with R*

This textbooks has a chapter on Bayesian analysis that we will use at the end of the semester.

Assignments

- DAACS (5%)
- Participation (5%)
  - One Minute Papers
- Homework (20%)
- Labs (40%)
  - Labs are designed to introduce to you doing statistics with R.
  - Answer the questions in the main text as well as the "On Your Own" section.
- Data Project (20%)
  - This allows you to analyze a dataset of your choosing. Projects will be shared with the class. This provides an opportunity for everyone to see different approaches to analyzing different datasets.
- Final exam (10%)

Communication

- Slack Channel: https://epsy630spring2021.slack.com
  - Click here to join the group
- Email: jbryer@albany.edu
- Phone/Zoom: Please email to schedule a time to meet.
- Office hours will typically be:
  - Fridays from 12:00am to 1:00pm
  - I will use the same Zoom link that we use for the Tuesday night meetups.

Familiarity with Statistical Topics

```
likert(stats.results) %>% plot(center = 2.5)
```

Math Anxiety Survey Scale

```
likert(mass.results) %>% plot()
```

Software

This is an applied statistics course so we will make extensive use of the R statistical programming language. You have two options for using R in this course:

- Use the R Studio server available here: r.bryer.org. I will email everyone a username and password (if you haven't received it, let me know).
- Install R and RStudio on your own computer. I encourage everyone to do this at some point by the end of the semester. I have instructions on the course website here: https://spring2021.data606.net/course-overview/software/

You will also need to have LaTeX installed as well in order to create PDFs. The `tinytex` R package helps with this process:

```
install.packages('tinytex')
tinytex::install_tinytex()
```

What is R?

"R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues..."

"R provides a wide variety of statistical (linear and non linear modeling, classical statistical tests, time-series analysis, classification, clustering, ...) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity."
(R-project.org)

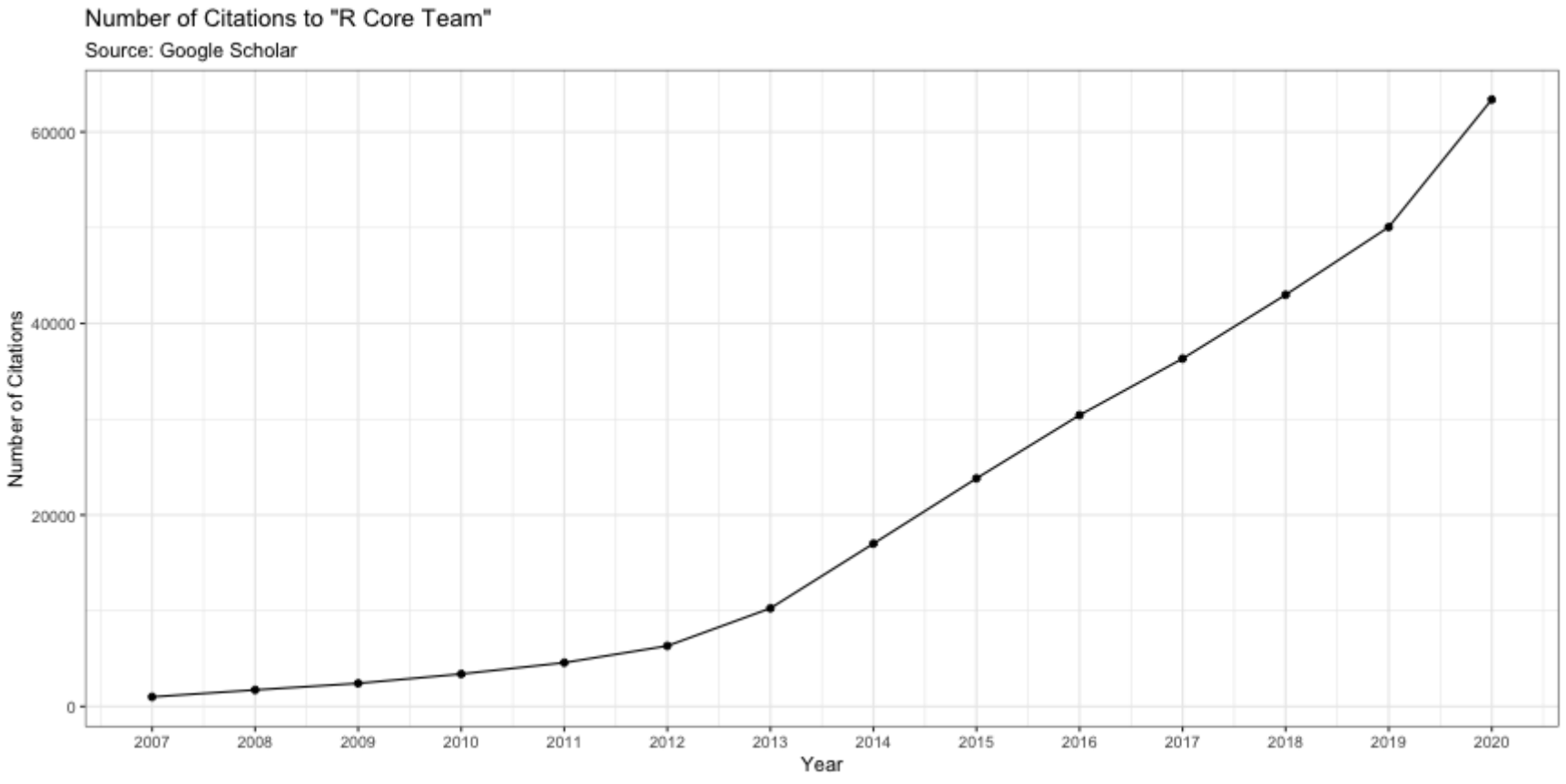
Pros

- FREE! *R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.*
- Available for multiple platforms (i.e. Windows, Mac, Linux).
- Easily extensible with (currently) 17,037 packages listed on CRAN.
- Scriptable.
- Publication grade graphics.
- Multiple ways of doing the same thing.
- Quickly becoming the *de facto* standard among statistician.

Cons

- Has a steeper learning curve.
- Multiple ways of doing the same thing.
- Can have difficulty with *very* large datasets.

The Popularity of R



Firth, D (2011). R and citations. Weblog entry at URL https://statgeek.wordpress.com/2011/06/25/r-and-citations/.

See also: Muenchen, R.A. (2017). The Popularity of Data Analysis Software. Welog entry at URL http://r4stats.com/articles/popularity/

Getting Started with RStudio

Go to https://r.bryer.org. You should have received an email with your username and password.

Once you login in, click the `Terminal` tab and type `passwd` to change your password.

Click back to the `Console` tab. This is where you type R commands.

R Markdown

R Markdown are a special type of document that allows you to combine your text with analysis. They are plain text documents where R code can be used to analyze and generate output (e.g. tables, figures) within the document. Text formatting uses simple markup called Markdown. You can then render the document to many output formats including MS Word, HTML, PDF, PowerPoint, HTML Slides (like this slide deck), and many more.

R Packages

One aspect that makes R popular is how (relatively) easy it is to extend it's functionality vis-à-vis R packages. R packages are collections of R functions, data, and documentation.

The Comprehensive R Archive Network (CRAN) is the central repository where R packages are published. However, it should be noted that there are mirrors located across the globe.

Using packages requires two steps: first, install the package (required once per R installation); and second, load the package (once per R session).

```
install.packages('likert')
```

```
library(likert)
```

Suggested Packages

These are the packages we will make frequent use during the semester. If you use the hosted version of R Studio these packages will already be installed.

```
pkgs <- c('tidyverse','devtools','reshape2','RSQLite',
          'psa','multilevelPSA','PSAboot','TriMatch','likert',
          'openintro','OIdata','psych','knitr','markdown','rmarkdown','shiny')

install.packages(pkgs)

devtools::install_github('jbryer/ipeds')
devtools::install_github('jbryer/sqlutils')
devtools::install_github("seankross/lego")
devtools::install_github("jbryer/DATA606")
```

DATA 606 Package

The `DATA606` R package contains many data sets and functions we will use throughout the semester. It also has a `startLab` function that will copy each of the labs to your current working directory. Use the following commands to install the package (only necessary once per R installation):

```
remotes::install_github('jbryer/DATA606')
```

To start the first lab...

```
DATA606::startLab('Lab1')
```

This will copy the R markdown file and any supporting files to your current working directory. Use the "Knit" button in R Studio to build a PDF of the document.

R as a Big Calculator

```
2 + 2
```

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

```
1 + sin(9)
```

```
## [1] 1.412118
```

```
exp(1) ^ (1i * pi)
```

```
## [1] -1+0i
```

Euler's Formula

$$e^{i\pi} + 1 = 0$$

"The most remarkable formula in mathematics"

- Richard Feyneman

Getting Help

R provides extensive documentation and help. The help.start() function will launch a webpage with links to:

- The R manuals
- The R FAQ
- Search engine
- and many other useful sites

The help.search() function will search the help file for a particular word or phrase. For example:

```
help.search('cross tabs')
```

To get documentation on a specific function, the `help()` function, or simply `?functionName` will open the documentation page in the web browser.

Lastly, to search the R mailing lists, use the `RSiteSearch()` function.

Reading Data

Data File Type	Extension	Function
R Data	rda, rdata	`base::load`, `base::readRDS`
Comma separated values	csv	`utils::read.csv`, `readr::read_csv`
Other delimited files		`utils::read.table`, `readr::read_delim`
Tab separated files		`readr::read_tsv`
Fixed width files		`utils::read.fwf`, `readr::read_fwf`
SPSS	sav	`foreign::read.spss`, `haven::read_sav`, `haven::read_por`
SAS	sas	`haven::read_sas`
Read lines		`base::scan`, `readr::read_lines`
Microsoft Excel	xls, xlsx	`gdata::read.xls`, `readxl::read_excel`

The R Language: Arithmetic Operators

- `+` - addition
- `-` - subtraction
- `*` - multiplication
- `/` - division
- `^` or `**` - exponentiation
- `x %% y` - modulus (x mod y) 5%%2 is 1

```
5 %% 2
```

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

- `x %/% y` - integer division

```
5%/%2
```

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

R Primitive Vectors

- `logicial` (e.g. `TRUE`, `FALSE`)
- `integer` - whole numbers, either positive or negative (e.g. `2112`, `42`, `-1`)
- `double` or `numeric` - real number (e.g. `0.05`, `pi`, `-Inf`, `NaN`)
- `complex` - complex number (e.g. `1i`)
- `character` - sequence of characters, or a string (e.g. `"Hello EPSY887!"`)

You can use the `class` function to determine the type of an object.

```
tmp <- c(2112, pi)
class(tmp)
```

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

R Primitive Vectors (cont.)

To test if an object is of a particular class, use the `is.XXX` set of functions:

```
is.double(tmp)
```

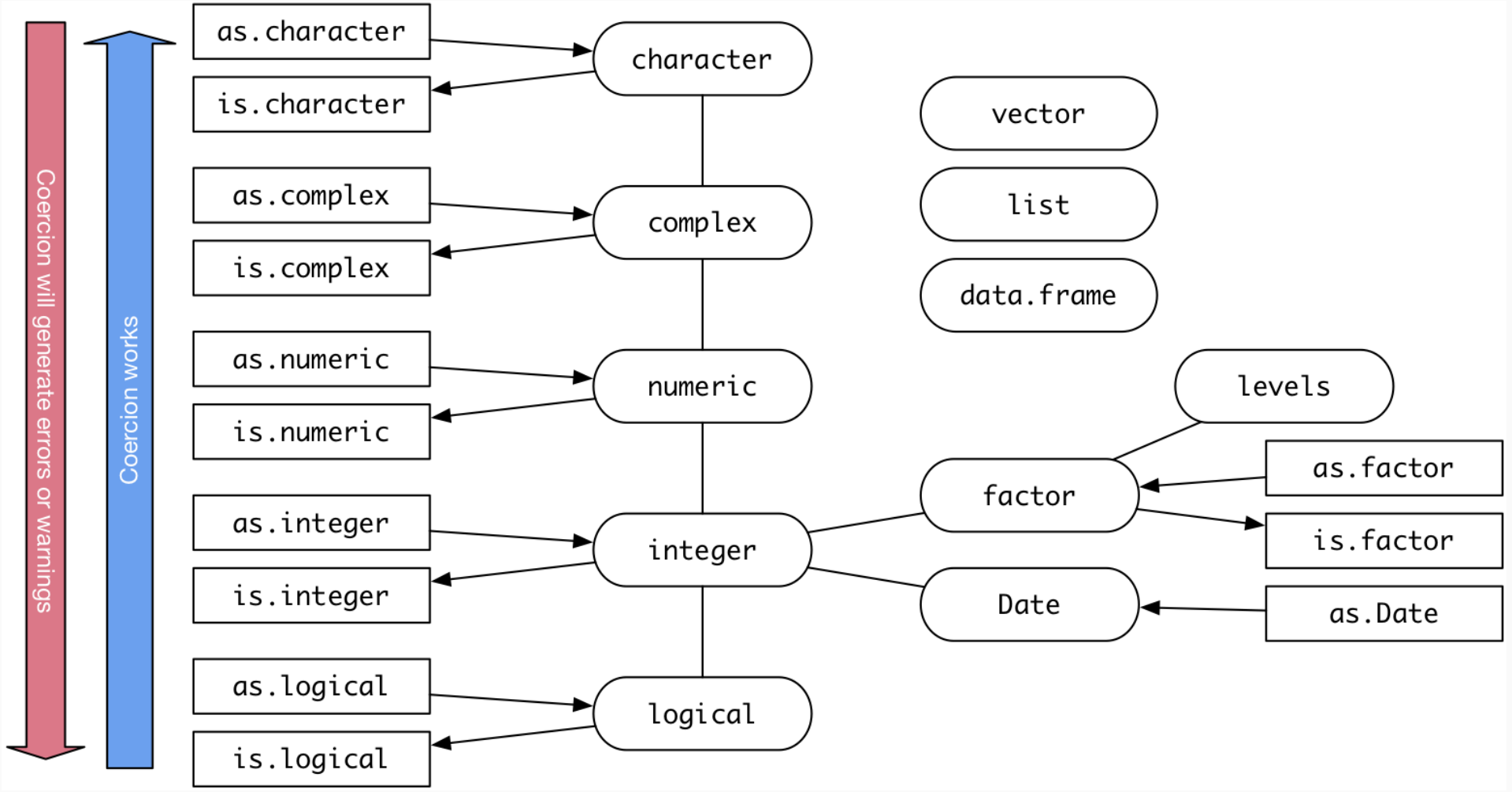
```
## [1] TRUE
```

And to convert from one type to another, use the `as.XXX` set of functions:

```
as.character(tmp)
```

```
## [1] "2112"              "3.14159265358979"
```

Data Types in R



Lists

A `list` is an object that contains a list of named values

```
tmp <- list(a = 2112, b = pi, z = "Hello EPS
tmp
```

```
## $a
## [1] 2112
##
## $b
## [1] 3.141593
##
## $z
## [1] "Hello EPSY887!"
```

```
tmp[1]; class(tmp[1]) # One square backet: r
```

```
## $a
## [1] 2112
```

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

```
tmp[[1]]; class(tmp[[2]]) # Two square brack
```

```
## [1] 2112
```

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

Factors

A `factor` is a way for R to store a nominal, or categorical, variable. R stores the underlying data as an integer where each value corresponds to a label.

```
gender <- c(rep("male",4), rep("female", 6))
gender
```

```
##  [1] "male"   "male"   "male"   "male"   "female" "female" "female" "female"
##  [9] "female" "female"
```

```
gender <- factor(gender, levels=c('male','female','unknown'))
gender
```

```
##  [1] male   male   male   male   female female female female female female
## Levels: male female unknown
```

```
levels(gender)
```

Factors can be ordered

The `ordered` parameter indicates whether the levels in the factor should be ordered.

```
library(TriMatch)
data(tutoring, package='TriMatch')
head(tutoring$Grade)
```

```
## [1] 4 4 4 4 4 3
```

```
grade <- tutoring$Grade
table(grade, useNA='ifany')
```

```
## grade
##   0   1   2   3   4
## 187  25  86 271 573
```

```
grade <- factor(tutoring$Grade,
```

```
table(grade, useNA='ifany')
```

```
## grade
##   F   D   C   B   A
## 187  25  86 271 573
```

With an ordered factor, coercing it back to an integer will maintain the order, but the values start with one!

```
head(grade)
```

```
## [1] A A A A A B
## Levels: F < D < C < B < A
```

Dates

R stores dates in `YYYY-MM-DD` format. The `as.Date` function will convert characters to `Date`s if they are in that form. If not, the `format` can be specified to help R coerce it to a `Date` format.

```
today <- Sys.Date()
format(today, '%B %d, $Y')
```

```
## [1] "February 02, $Y"
```

```
as.Date('2015-NOV-01', format='%Y-%b-%d')
```

```
## [1] "2015-11-01"
```

- `%d` - day as a number (i.e 0-31)
- `%a` - abbreviated weekday (e.g. `Mon`)
- `%A` - unabbreviated weekday (e.g. `Monday`)
- `%m` - month (i.e. 00-12)
- `%b` - abbreviated month (e.g. `Jan`)
- `%B` - unabbreviated month (e.g. `January`)
- `%y` - 2-digit year (e.g. `15`)
- `%Y` - 4-digit year (e.g. `2015`)

NA versus NULL

R is just as much a programming language as it is a statistical software package. As such it represents null differently for programming (using `NULL`) than for data (using `NA`).

- `NULL` represents the null object in R: it is a reserved word. NULL is often returned by expressions and functions whose values are undefined.

- `NA` is a logical constant of length 1 which contains a missing value indicator. `NA` can be freely coerced to any other vector type except raw. There are also constants `NA_integer`, `NA_real`, `NA_complex`, and `NA_character` of the other atomic vector types which support missing values: all of these are reserved words in the R language.

For more details, see http://opendatagroup.com/2010/04/25/r-na-v-null/

Handling Missing Data

There are a number of functions available for finding and subsetting missing values:

- `is.na` - function that takes one parameter and returns a logical vector of the same length where `TRUE` indicates the value is missing in the original vector.
- `complete.cases` - function that takes a data frame or matrix and returns `TRUE` if the entire row has *no* missing values.
- `na.omit` - function that takes a data frame and matrix and returns a subset of that data frame or matrix with any rows containing missing values removed.

Many statistical functions (e.g. `mean`, `sd`, `cor`) have a `na.rm` parameter that, when `TRUE`, will remove any missing values before calculating the statistic.

There are two very good R packages for imputing missing values:

- `mice` - Multivariate Imputation by Chained Equations
- `Amelia II` - A Program for Missing Data

Data Frames

Data frames are collection of vectors, thereby making them two dimensional. Unlike matrices (see `?matrix`) where all data elements are of the same type (i.e. numeric, character, logical, complex), each column in a data frame can be of a different type.

```
class(mass.results)
```

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

```
dim(mass.results) # Dimension of the data frame (row by column)
```

```
## [1] 19 14
```

```
nrow(mass.results) # Number of rows
```

```
## [1] 19
```

str

The `str` is perhaps the most useful function in R. It displays the structure of an R object.

```
str(mass.results)
```

```
## 'data.frame':    19 obs. of  14 variables:
##  $ I find math interesting.                                      : Ord.factor w/
##  $ I get uptight during math tests.                              : Ord.factor w/
##  $ I think that I will use math in the future.                   : Ord.factor w/
##  $ Mind goes blank and I am unable to think clearly when doing my math test.: Ord.factor w/
##  $ Math relates to my life.                                      : Ord.factor w/
##  $ I worry about my ability to solve math problems.              : Ord.factor w/
##  $ I get a sinking feeling when I try to do math problems.       : Ord.factor w/
##  $ I find math challenging.                                      : Ord.factor w/
##  $ Mathematics makes me feel nervous.                            : Ord.factor w/
##  $ I would like to take more math classes.                       : Ord.factor w/
##  $ Mathematics makes me feel uneasy.                             : Ord.factor w/
##  $ Math is one of my favorite subjects.                          : Ord.factor w/
##  $ I enjoy learning with mathematics.                            : Ord.factor w/
```

Exploring the Data in Data Frames

```
head(mass.results)
```

```
##   I find math interesting. I get uptight during math tests.
## 1                  Disagree                  Strongly Disagree
## 2           Strongly Agree                              Agree
## 3                  Neutral                              Agree
## 4                    Agree                            Neutral
## 5         Strongly Disagree                              Agree
## 6                    Agree                           Disagree
##   I think that I will use math in the future.
## 1                                     Neutral
## 2                                       Agree
## 3                              Strongly Agree
## 4                              Strongly Agree
## 5                                       Agree
## 6                                    Disagree
##   Mind goes blank and I am unable to think clearly when doing my math test.
## 1                                                                   Neutral
```

Subsetting Data Frames

Using square brackets will allow you to subset from a data frame. The first parameter is for rows, the second for columns. Leaving one blank will return all rows or columns.

```
mass.results[c(1:2,10),] # Return the first, second, and tenth row
```

```
##    I find math interesting. I get uptight during math tests.
## 1                   Disagree                  Strongly Disagree
## 2             Strongly Agree                              Agree
## 10                    Agree                           Disagree
##    I think that I will use math in the future.
## 1                                      Neutral
## 2                                        Agree
## 10                                       Agree
##    Mind goes blank and I am unable to think clearly when doing my math test.
## 1                                                                    Neutral
## 2                                                                   Disagree
## 10                                                                  Disagree
##    Math relates to my life. I worry about my ability to solve math problems.
```

Subsetting Missing Values

Using the `complete.cases` function, we can return rows with at least one missing values.

```
mass.results[!complete.cases(mass.results),]
```

```
##  [1] I find math interesting.
##  [2] I get uptight during math tests.
##  [3] I think that I will use math in the future.
##  [4] Mind goes blank and I am unable to think clearly when doing my math test.
##  [5] Math relates to my life.
##  [6] I worry about my ability to solve math problems.
##  [7] I get a sinking feeling when I try to do math problems.
##  [8] I find math challenging.
##  [9] Mathematics makes me feel nervous.
## [10] I would like to take more math classes.
## [11] Mathematics makes me feel uneasy.
## [12] Math is one of my favorite subjects.
## [13] I enjoy learning with mathematics.
## [14] Mathematics makes me feel confused.
```

Subsetting Missing Values

Using the `is.na`, we can change replace the missing values.

```
(tmp <- sample(c(1:5, NA)))
```

```
## [1]  4  2  3  5  1 NA
```

```
tmp[is.na(tmp)] <- 2112
tmp
```

```
## [1]    4    2    3    5    1 2112
```

Tip: One Column Data Frame

When selecting one column from a data frame, R will convert the returned object to a vector.

```
class(mass.results[,1])
```

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

You can use the `drop=FALSE` parameter keep the subset as a data frame.

```
class(mass.results[,1,drop=FALSE])
```

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

Subsetting with Logical Operators

You can subset using logical vectors. For example, there are 1142 rows in the `tutoring` data frame loaded from the `TriMatch` package You can pass a logical vector of length 1142 where `TRUE` indicates to return the row and `FALSE` to not. For example, we wish to return the rows where students received an F:

```
row <- tutoring$GradeCode == 'F'
length(row)
```

```
## [1] 1142
```

Here we are using the `==` logical operator. This will test each element in the `tutoring$GradeCode` and return `TRUE` if it is equal to `F`, `FALSE` otherwise.

```
tutoring[row, ]
```

```
##       treat  Course Grade Gender Ethnicity Military   ESL EdMother EdFather
```

which

The **which** command will return an **integer** vector with the positions within the **logical** vector that are **TRUE**.

```
which(row)
```

```
##  [1]   17   22   49   57   59   60   67   90  119  144  149  180  191  217  230
## [16]  264  301  306  325  386  404  443  469  504  505  520  521  530  534  564
## [31]  565  568  572  583  588  613  632  640  656  664  666  671  679  681  689
## [46]  720  724  731  743  809  828  833  863  897  962  999 1005 1045 1081 1103
## [61] 1112 1117 1120 1140
```

```
tutoring[17, 1:16]
```

```
##      treat  Course Grade Gender Ethnicity Military   ESL EdMother EdFather Age
## 40 Control ENG*201     0 FEMALE     White     TRUE FALSE        5        6  42
##    Employment Income Transfer  GPA GradeCode Level
## 40          3      5    36.63 2.36         F Lower
```

Logical Operators

- `!a` - TRUE if a is FALSE
- `a == b` - TRUE if a and be are equal
- `a != b` - TRUE if a and b are not equal
- `a > b` - TRUE if a is larger than b, but not equal
- `a >= b` - TRUE if a is larger or equal to b
- `a < b` - TRUE if a is smaller than be, but not equal
- `a <= b` - TRUE if a is smaller or equal to b
- `a %in% b` - TRUE if a is in b where b is a vector

```
which( letters %in% c('a','e','i','o','u') )
```

```
## [1]  1  5  9 15 21
```

- `a | b` - TRUE if a *or* b are TRUE
- `a & b` - TRUE if a *and* b are TRUE
- `isTRUE(a)` - TRUE if a is TRUE

Side Note: Operators are Functions

All operations (e.g. `+`, `-`, `*`, `/`, `[`, `<-`) are functions.

```
class(`+`)
```

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

```
`+`
```

```
## function (e1, e2)  .Primitive("+")
```

```
`+`(2, 3)
```

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

You can redefine these functions, but probably not a good idea ;-)

Sorting Data

The `order` function will take one or more vectors (usually in the form of a data frame) and return an integer vector indicating the new order. There are two parameters to adjust where `NA`s are placed (`na.last=FALSE`) and whether to sort in increasing or decreasing order (`decreasing=FALSE`).

```
(randomLetters <- sample(letters))
```

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

```
randomLetters[order(randomLetters)]
```

```
##  [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"
```

```
randomLetters[order(randomLetters, decreasing=TRUE)]
```

Lab 1

Let's start working on Lab 1.

Login the RStudio server.

Run the following commands:

```
library(DATA606)
startLab('Lab1')
```

Click the **`Knit`** button to build the R Markdown file.

Complete the one minute paper: https://forms.gle/yB3ds6MYE89Z1pURA

1. What was the most important thing you learned during this class?
2. What important question remains unanswered for you?

Good luck with the semester!

jbryer@albany.edu
data606spring2021.slack.com
@jbryer
@jbryer
@jasonbryer
epsy630.bryer.org
bryer.org