UCSD Political Science Math Camp

Under Construction

Contents

About this Booklet

The UC San Diego Math camp is held each year in September. The 2021 Math Camp instructors are Keng-Chi Chang and Rachel Schoner. The faculty coordinator is Molly Roberts.

This booklet serves as the text for the Math Camp, available as a webpage updated automatically. Is it the product of generations of Math Camp instructors and is adapted from the Harvard Gov Prefresher.

For information about the role of this math camp as an introduction to graduate school, you may be interested in "The Math Prefresher and The Collective Future of Political Science Graduate Training, in PS: Political Science & Politics, by Gary King, Shiro Kuriwaki, and Yon Soo Park.

Authors and Contributors

This material is adapted from previous instructors of the class at UCSD and Harvard Gov Prefresher.

- Past Authors and Instructors:
- Repository Maintainer:

Contributing

We transitioned the booklet into a bookdown github repository in 2021. As we update this version, we appreciate any bug reports or fixes appreciated.

All changes should be made in the `.Rmd` files in the project root. To contribute a change, please make a pull request and set the repository maintainer as the reviewer.

Pre-Math Camp Exercises

Before our first meeting, please try solving these questions. They are a sample of the very beginning of each math section. We have provided links to the parts of the book you can read if the concepts are new to you.

The goal of this "pre"-math camp assignment is not to intimidate you but to set common expectations so you can make the most out of the actual Math Camp. Even if you do not understand some or all of these questions after skimming through the linked sections, your effort will pay off, and you will be better prepared for the math camp. We are also open to adjusting these expectations based on feedback (this class is for *you*), so please do not hesitate to write to the instructors for feedback.

Linear Algebra

Vectors

Define the vectors $u = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$, $v = \begin{pmatrix} 4 \\ 5 \\ 6 \end{pmatrix}$, and the scalar $c = 2$. Calculate the following:

1. $u + v$
2. cv
3. $u \cdot v$

If you are having trouble with these problems, please review Section **??** "Working with Vectors" in Chapter **??**.

Are the following sets of vectors linearly independent?

1. $u = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$, $v = \begin{pmatrix} 2 \\ 4 \end{pmatrix}$
2. $u = \begin{pmatrix} 1 \\ 2 \\ 5 \end{pmatrix}$, $v = \begin{pmatrix} 3 \\ 7 \\ 9 \end{pmatrix}$
3. $a = \begin{pmatrix} 2 \\ -1 \\ 1 \end{pmatrix}$, $b = \begin{pmatrix} 3 \\ -4 \\ -2 \end{pmatrix}$, $c = \begin{pmatrix} 5 \\ -10 \\ -8 \end{pmatrix}$ (this requires some guesswork)

If you are having trouble with these problems, please review Section **??**.

Matrices

$$= \begin{pmatrix} 7 & 5 & 1 \\ 11 & 9 & 3 \\ 2 & 14 & 21 \\ 4 & 1 & 5 \end{pmatrix}$$

What is the dimensionality of matrix ?

What is the element a_{23} of ?

Given that

$$= \begin{pmatrix} 1 & 2 & 8 \\ 3 & 9 & 11 \\ 4 & 7 & 5 \\ 5 & 1 & 9 \end{pmatrix}$$

What is $+$?

Given that

$$= \begin{pmatrix} 1 & 2 & 8 \\ 3 & 9 & 11 \\ 4 & 7 & 5 \end{pmatrix}$$

What is $+$?

Given that

$$c = 2$$

What is c?

If you are having trouble with these problems, please review Section **??**.

Operations

Summation

Simplify the following

1. $\sum_{i=1}^{3} i$
2. $\sum_{k=1}^{3} (3k + 2)$

3. $\sum_{i=1}^{4}(3k+i+2)$

Products

1. $\prod_{i=1}^{3} i$
2. $\prod_{k=1}^{3}(3k+2)$

To review this material, please see Section **??**.

Logs and exponents

Simplify the following

1. 4^2
2. $4^2 2^3$
3. $\log_{10} 100$
4. $\log_2 4$
5. $\log e$, where log is the natural log (also written as ln) – a log with base e, and e is Euler's constant
6. $e^a e^b e^c$, where a, b, c are each constants
7. $\log 0$
8. e^0
9. e^1
10. $\log e^2$

To review this material, please see Section **??**

Limits

Find the limit of the following.

1. $\lim_{x \to 2}(x-1)$
2. $\lim_{x \to 2} \frac{(x-2)(x-1)}{(x-2)}$
3. $\lim_{x \to 2} \frac{x^2-3x+2}{x-2}$

To review this material please see Section **??**

Calculus

For each of the following functions $f(x)$, find the derivative $f'(x)$ or $\frac{d}{dx}f(x)$

1. $f(x) = c$
2. $f(x) = x$

3. $f(x) = x^2$
4. $f(x) = x^3$
5. $f(x) = 3x^2 + 2x^{1/3}$
6. $f(x) = (x^3)(2x^4)$

For a review, please see Section **??** - **??**

Optimization

For each of the followng functions $f(x)$, does a maximum and minimum exist in the domain $x \in$? If so, for what are those values and for which values of x?

1. $f(x) = x$
2. $f(x) = x^2$
3. $f(x) = -(x-2)^2$

If you are stuck, please try sketching out a picture of each of the functions.

Probability

1. If there are 12 cards, numbered 1 to 12, and 4 cards are chosen, how many distinct possible choices are there? (unordered, without replacement)
2. Let $A = \{1, 3, 5, 7, 8\}$ and $B = \{2, 4, 7, 8, 12, 13\}$. What is $A \cup B$? What is $A \cap B$? If A is a subset of the Sample Space $S = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, what is the complement A^C?
3. If we roll two fair dice, what is the probability that their sum would be 11?
4. If we roll two fair dice, what is the probability that their sum would be 12?

For a review, please see Sections **??** - **??**.

Chapter 1

Preliminaries

Topics Dimensionality; Interval Notation for ^{1}; Neighborhoods: Intervals, Disks, and Balls; Introduction to Functions; Domain and Range; Some General Types of Functions; log, ln, and exp; Other Useful Functions; Graphing Functions; Solving for Variables; Finding Roots; Limit of a Function; Continuity; Sets, Sets, and More Sets.

1.1 Summation Operators

Addition (+), Subtraction (-), multiplication and division are basic operations of arithmetic – combining numbers. In statistics and calculus, we want to add a *sequence* of numbers that can be expressed as a pattern without needing to write down all its components. For example, how would we express the sum of all numbers from 1 to 100 without writing a hundred numbers?

For this we use the summation operator $\sum$ and the product operator $\prod$.

Summation:

$$\sum_{i=1}^{100} x_i = x_1 + x_2 + x_3 + \cdots + x_{100}$$

The bottom of the $\sum$ symbol indicates an index (here, i), and its start value 1. At the top is where the index ends. The notion of "addition" is part of the $\sum$ symbol. The content to the right of the summation is the meat of what we add. While you can pick your favorite index, start, and end values, the content must also have the index.

- $\sum\limits_{i=1}^{n} cx_i = c\sum\limits_{i=1}^{n} x_i$
- $\sum\limits_{i=1}^{n} (x_i + y_i) = \sum\limits_{i=1}^{n} x_i + \sum\limits_{i=1}^{n} y_i$
- $\sum\limits_{i=1}^{n} c = nc$

Product:

$$\prod_{i=1}^{n} x_i = x_1 x_2 x_3 \cdots x_n$$

Properties:

- $\prod\limits_{i=1}^{n} cx_i = c^n \prod\limits_{i=1}^{n} x_i$
- $\prod\limits_{i=k}^{n} cx_i = c^{n-k+1} \prod\limits_{i=k}^{n} x_i$
- $\prod\limits_{i=1}^{n} (x_i + y_i) =$ a total mess
- $\prod\limits_{i=1}^{n} c = c^n$

Other Useful Functions

Factorials!:

$$x! = x \cdot (x-1) \cdot (x-2) \cdots (1)$$

Modulo: Tells you the remainder when you divide the first number by the second.

- $17 \mod 3 = 2$
- $100 \ \% \ 30 = 10$

Example 1.1 (Operators). 1. $\sum\limits_{i=1}^{5} i =$

2. $\prod\limits_{i=1}^{5} i =$

3. $14 \mod 4 =$

4. $4! =$

Exercise 1.1 (Operators). Let $x_1 = 4, x_2 = 3, x_3 = 7, x_4 = 11, x_5 = 2$

1. $\sum\limits_{i=1}^{3} (7)x_i$

2. $\sum\limits_{i=1}^{5} 2$

3. $\prod\limits_{i=3}^{5} (2)x_i$

1.2 Introduction to Functions

A **function** (in ^{1}) is a mapping, or transformation, that relates members of one set to members of another set. For instance, if you have two sets: set A and set B, a function from A to B maps every value a in set A such that $f(a) \in B$. Functions can be "many-to-one", where many values or combinations of values from set A produce a single output in set B, or they can be "one-to-one", where each value in set A corresponds to a single value in set B. A function by definition has a single function value for each element of its domain. This means, there cannot be "one-to-many" mapping.

Dimensionality: ^{1} is the set of all real numbers extending from $-\infty$ to $+\infty$ — i.e., the real number line. ^{n} is an n-dimensional space, where each of the n axes extends from $-\infty$ to $+\infty$.

- ^{1} is a one dimensional line.
- ^{2} is a two dimensional plane.
- ^{3} is a three dimensional space.

Points in ^{n} are ordered n-tuples (just means an combination of n elements where order matters), where each element of the n-tuple represents the coordinate along that dimension.

For example:

- ^{1}: (3)
- ^{2}: (-15, 5)
- ^{3}: (86, 4, 0)

Examples of mapping notation:

Function of one variable: $f : {}^1 \to {}^1$

- $f(x) = x + 1$. For each x in ^{1}, $f(x)$ assigns the number $x + 1$.

Function of two variables: $f : {}^2 \to {}^1$.

- $f(x, y) = x^2 + y^2$. For each ordered pair (x, y) in ^{2}, $f(x, y)$ assigns the number $x^2 + y^2$.

We often use variable x as input and another y as output, e.g. $y = x + 1$

Example 1.2 (Functions)**.** For each of the following, state whether they are one-to-one or many-to-one functions.

1. For $x \in [0, \infty]$, $f : x \to x^2$ (this could also be written as $f(x) = x^2$).
2. For $x \in [-\infty, \infty]$, $f : x \to x^2$.

Exercise 1.2 (Functions)**.** For each of the following, state whether they are one-to-one or many-to-one functions.

1. For $x \in [-3, \infty]$, $f : x \to x^2$.
2. For $x \in [0, \infty]$, $f : x \to \sqrt{x}$

Some functions are defined only on proper subsets of ^{n}.

- **Domain**: the set of numbers in X at which $f(x)$ is defined.

- **Range**: elements of Y assigned by $f(x)$ to elements of X, or
$$f(X) = \{y : y = f(x), x \in X\}$$
Most often used when talking about a function $f : {}^1 \to {}^1$.
- **Image**: same as range, but more often used when talking about a function $f : {}^n \to {}^1$.

Some General Types of Functions

Monomials: $f(x) = ax^k$

a is the coefficient. k is the degree.

Examples: $y = x^2$, $y = -\frac{1}{2}x^3$

Polynomials: sum of monomials.

Examples: $y = -\frac{1}{2}x^3 + x^2$, $y = 3x + 5$

The degree of a polynomial is the highest degree of its monomial terms. Also, it's often a good idea to write polynomials with terms in decreasing degree.

Exponential Functions: Example: $y = 2^x$

1.3 log and exp

Relationship of logarithmic and exponential functions:

$$y = \log_a(x) \iff a^y = x$$

The log function can be thought of as an inverse for exponential functions. a is referred to as the "base" of the logarithm.

Common Bases: The two most common logarithms are base 10 and base e.

1. Base 10: $y = \log_{10}(x) \iff 10^y = x$. The base 10 logarithm is often simply written as "log(x)" with no base denoted.
2. Base e: $y = \log_e(x) \iff e^y = x$. The base e logarithm is referred to as the "natural" logarithm and is written as "ln(x)".

Properties of exponential functions:

- $a^x a^y = a^{x+y}$
- $a^{-x} = 1/a^x$
- $a^x/a^y = a^{x-y}$
- $(a^x)^y = a^{xy}$
- $a^0 = 1$

Properties of logarithmic functions (any base):

Generally, when statisticians or social scientists write $\log(x)$ they mean $\log_e(x)$. In other words: $\log_e(x) \equiv \ln(x) \equiv \log(x)$

$$\log_a(a^x) = x$$

and

$$a^{\log_a(x)} = x$$

- $\log(xy) = \log(x) + \log(y)$
- $\log(x^y) = y\log(x)$
- $\log(1/x) = \log(x^{-1}) = -\log(x)$
- $\log(x/y) = \log(x \cdot y^{-1}) = \log(x) + \log(y^{-1}) = \log(x) - \log(y)$
- $\log(1) = \log(e^0) = 0$

Change of Base Formula: Use the change of base formula to switch bases as necessary:

$$\log_b(x) = \frac{\log_a(x)}{\log_a(b)}$$

Example:

$$\log_{10}(x) = \frac{\ln(x)}{\ln(10)}$$

You can use logs to go between sum and product notation. This will be particularly important when you're learning maximum likelihood estimation.

$$\begin{aligned}
\log\left(\prod_{i=1}^{n} x_i\right) &= \log(x_1 \cdot x_2 \cdot x_3 \cdots \cdot x_n) \\
&= \log(x_1) + \log(x_2) + \log(x_3) + \cdots + \log(x_n) \\
&= \sum_{i=1}^{n} \log(x_i)
\end{aligned}$$

Therefore, you can see that the log of a product is equal to the sum of the logs. We can write this more generally by adding in a constant, c:

$$\begin{aligned}
\log\left(\prod_{i=1}^{n} cx_i\right) &= \log(cx_1 \cdot cx_2 \cdots cx_n) \\
&= \log(c^n \cdot x_1 \cdot x_2 \cdots x_n) \\
&= \log(c^n) + \log(x_1) + \log(x_2) + \cdots + \log(x_n) \\
&= n\log(c) + \sum_{i=1}^{n} \log(x_i)
\end{aligned}$$

Example 1.3 (Logarithmic Functions)**.** Evaluate each of the following logarithms

1. $\log_4(16)$
2. $\log_2(16)$

Simplify the following logarithm. By "simplify", we actually really mean - use as many of the logarithmic properties as you can.

3. $\log_4(x^3y^5)$

Exercise 1.3 (Logarithmic Functions)**.** Evaluate each of the following logarithms

1. $\log_{\frac{3}{2}}(\frac{27}{8})$

Simplify each of the following logarithms. By "simplify", we actually really mean - use as many of the logarithmic properties as you can.

2. $\log(\frac{x^9y^5}{z^3})$
3. $\ln\sqrt{xy}$

1.4 Graphing Functions

What can a graph tell you about a function?

- Is the function increasing or decreasing? Over what part of the domain?
- How "fast" does it increase or decrease?
- Are there global or local maxima and minima? Where?
- Are there inflection points?
- Is the function continuous?
- Is the function differentiable?
- Does the function tend to some limit?
- Other questions related to the substance of the problem at hand.

1.5 Solving for Variables and Finding Roots

Sometimes we're given a function $y = f(x)$ and we want to find how x varies as a function of y. Use algebra to move x to the left hand side (LHS) of the equation and so that the right hand side (RHS) is only a function of y.

Example 1.4 (Solving for Variables)**.** Solve for x:

1. $y = 3x + 2$
2. $y = e^x$

Solving for variables is especially important when we want to find the **roots** of an equation: those values of variables that cause an equation to equal zero. Especially important in finding equilibria and in doing maximum likelihood estimation.

Procedure: Given $y = f(x)$, set $f(x) = 0$. Solve for x.

Multiple Roots:

$$f(x) = x^2 - 9 \quad \Longrightarrow \quad 0 = x^2 - 9 \quad \Longrightarrow \quad 9 = x^2 \quad \Longrightarrow \quad \pm\sqrt{9} = \sqrt{x^2} \quad \Longrightarrow \quad \pm 3 = x$$

Quadratic Formula: For quadratic equations $ax^2 + bx + c = 0$, use the quadratic formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Exercise 1.4 (Finding Roots)**.** Solve for x:

1. $f(x) = 3x + 2 = 0$
2. $f(x) = x^2 + 3x - 4 = 0$
3. $f(x) = e^{-x} - 10 = 0$

1.6 Sets

Interior Point: The point is an interior point of the set S if is in S and if there is some ϵ-ball around that contains only points in S. The **interior** of S is the collection of all interior points in S. The interior can also be defined as the union of all open sets in S.

- If the set S is circular, the interior points are everything inside of the circle, but not on the circle's rim.
- Example: The interior of the set $\{(x, y) : x^2 + y^2 \leq 4\}$ is $\{(x, y) : x^2 + y^2 < 4\}$.

Boundary Point: The point is a boundary point of the set S if every ϵ-ball around contains both points that are in S and points that are outside S. The **boundary** is the collection of all boundary points.

- If the set S is circular, the boundary points are everything on the circle's rim.
- Example: The boundary of $\{(x, y) : x^2 + y^2 \leq 4\}$ is $\{(x, y) : x^2 + y^2 = 4\}$.

Open: A set S is open if for each point in S, there exists an open ϵ-ball around completely contained in S.

- If the set S is circular and open, the points contained within the set get infinitely close to the circle's rim, but do not touch it.
- Example: $\{(x, y) : x^2 + y^2 < 4\}$

Closed: A set S is closed if it contains all of its boundary points.

- Alternatively: A set is closed if its complement is open.
- If the set S is circular and closed, the set contains all points within the rim as well as the rim itself.
- Example: $\{(x, y) : x^2 + y^2 \leq 4\}$
- Note: a set may be neither open nor closed. Example: $\{(x, y) : 2 < x^2 + y^2 \leq 4\}$

Complement: The complement of set S is everything outside of S.

- If the set S is circular, the complement of S is everything outside of the circle.
- Example: The complement of $\{(x, y) : x^2 + y^2 \leq 4\}$ is $\{(x, y) : x^2 + y^2 > 4\}$.

Empty: The empty (or null) set is a unique set that has no elements, denoted by $\{\}$ or $\emptyset$.

- The empty set is an example of a set that is open and closed, or a "clopen" set.
- Examples: The set of squares with 5 sides; the set of countries south of the South Pole.

Answers to Examples and Exercises

Answer to Example **??**:

1. $1 + 2 + 3 + 4 + 5 = 15$
2. $1 * 2 * 3 * 4 * 5 = 120$
3. 2
4. $4 * 3 * 2 * 1 = 24$

Answer to Exercise **??**:

1. $7(4 + 3 + 7) = 98$
2. $2 + 2 + 2 + 2 + 2 = 10$
3. $2^3(7)(11)(2) = 1232$

Answer to Example **??**:

1. one-to-one
2. many-to-one

Answer to Exercise **??**:

1. many-to-one
2. one-to-one

Answer to Example **??**:

1. 2
2. 4
3. $3\log_4(x) + 5\log_4(y)$

Answer to Exercise **??**:

1. 3
2. $9\log(x) + 5\log(y) - 3\log(z)$
3. $\frac{1}{2}(\ln x + \ln y)$

Answer to Example **??**:

1. $y = 3x + 2 \quad \Longrightarrow \quad -3x = 2 - y \quad \Longrightarrow \quad 3x = y - 2 \quad \Longrightarrow \quad x = \frac{1}{3}(y - 2)$
2. $x = \ln y$

Answer to Exercise **??**:

1. $\frac{-2}{3}$
2. x = {1, -4}
3. x = - $\ln 10$

Part I

Introduction to R:

Chapter 2

Orientation and Reading in Data

Welcome to Math Camp! The first day we will get to know each other and get acquainted with R. We will be working with R throughout this course, and today is the beginning, where you will learn about this program. It can be overwhelming to learn any programming language, but we will spend a lot of time and practice with it. Please work through the following tutorials and exercises. Please come to class with areas of confusion and questions; we will walk through more examples together.

You should downloaded R and R Studio. This video has a nice brief introduction.

R Studio Cloud has great resources we will be using throughout this course. Please complete the following primer tutorials for the first session: * The Basics: Programming Basics

Motivation: Data and You

The modal social science project starts by importing existing datasets. Datasets come in all shapes and sizes. As you search for new data you may encounter dozens of file extensions – csv, xlsx, dta, sav, por, Rdata, Rds, txt, xml, json, shp … the list continues. Although these files can often be cumbersome, its a good to be able to find a way to encounter any file that your research may call for.

Reviewing data import will allow us to get on the same page on how computer systems work.

Where are we? Where are we headed?

Today we'll cover:

- What's what in RStudio
- What R is, at a high level
- How to read in data

- Comment on coding style on the way

Check your understanding

- What is the difference between a file and a folder?
- In the RStudio windows, what is the difference between the "Source" Pane and the "Console"? What is a "code chunk"?
- How do you read a R help page? What is the `Usage` section, the `Values` section, and the `Examples` section?
- What use is the "Environment" Pane?
- How would you read in a spreadsheet in R?
- How would you figure out what variables are in the data? size of the data?
- How would you read in a `csv` file, a `dta` file, a `sav` file?

2.1 Orienting

1. RStudio is a **GUI** and an IDE for the programming language R. A Graphical User Interface allows users to interface with the software (in this case R) using graphical aids like buttons and tabs. Often we don't think of GUIs because to most computer users, everything is a GUI (like Microsoft Word or your "Control Panel"), but it's always there! A Integrated Development Environment just says that the software to interface with R comes with useful useful bells and whistles to give you shortcuts.

   The **Console** is kind of a the core window through which you see your GUI actually operating through R. It's not graphical so might not be as intuitive. But all your results, commands, errors, warnings.. you see them in here. A console tells you what's going on now.

2. via the GUI, you the analyst needs to sends instructions, or **commands**, to the R application. The verb for this is "run" or "execute" the command. Computer programs ask users to provide instructions in very specific formats. While a English-speaking human can understand a sentence with a few typos in it by filling in the blanks, the same typo or misplaced character would halt a computer program. Each program has its own requirements for how commands should be typed; after all, each of these is its own language. We refer to the way a program needs its commands to be formatted as its **syntax**.

3. Theoretically, one could do all their work by typing in commands into the Console. But that would be a lot of work, because you'd have to give instructions each time you start your data analysis. Moreover, you'll have no record of what you did. That's why you need a **script**. This is a type of **code**. It can be referred to as a **source** because that is the source of your commands. Source is also used as a verb; "source the script" just means execute it. RStudio doesn't start out with a script, so you can make one from "File > New" or the New file icon.

4. You can also open scripts that are in folders in your computer. A script is a type of File. Find your Files in the bottom-right "Files" pane.

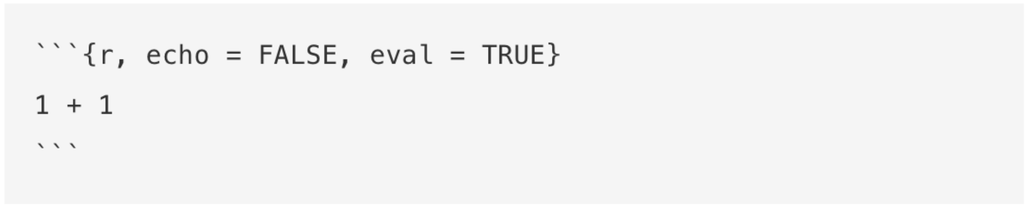
To load a dataset, you need to specify where that file is. Computer files (data, documents, programs) are organized hiearchically, like a branching tree. Folders can contain files, and also other folders. The GUI toolbar makes this lineaer and hiearchical relationship apparent. When we turn to locate the file in our commands, we need another set of syntax. Importantly, denote the hierarchy of a folder by the `/` (slash) symbol. `data/input/2018-08` indicates the `2018-08` folder, which is included in the `input` folder, which is in turn included in the `data` folder.

Files (but not folders) have "file extensions" which you are probably familiar with already: `.docx`, `.pdf`, and `.pdf`. The file extensions you will see in a stats or quantitative social science class are:

- `.pdf`: PDF, a convenient format to view documents and slides in, regardless of Mac/Windows.
- `.csv`: A comma separated values file
- `.xlsx`: Microsoft Excel file
- `.dta`: Stata data
- `.sav`: SPSS data
- `.R`: R code (script)
- `.Rmd`: Rmarkdown code (text + code)
- `.do`: Stata code (script)

5. In R, there are two main types of scripts. A classic `.R` file and a `.Rmd` file (for Rmarkdown). A .R file is just lines and lines of R code that is meant to be inserted right into the Console. A .Rmd tries to weave code and English together, to make it easier for users to create reports that interact with data and intersperse R code with explanation. For example, we built this book in Rmds.

```
The Rmarkdown facilitates is the use of __code chunks__, which are used here. These start and end with
```



~~~
```{r, echo = FALSE, eval = TRUE}
1 + 1
```
~~~

Figure 2.1: A code chunk in Rmarkdown (before rendering)

2.2 But what is R?

R is an object oriented programming language primarily used for statistical computing. An object oriented language is a programming language built around manipulating objects.

- In R, objects can be matrices, vectors, scalars, strings, and data frames, for example
- Objects contain different types of information
- Different objects have different allowable procedures:

```r
# Adding a string and a string does not work because the '+' operator
# does not work for strings:

# 'UCSD' + 'PoliSci'

# The '+' operator can add numbers just fine
9 + 13
```

```
## [1] 22
```

```r
# Reminder: to figure out the type of an object use:
x <- 9
class(9)
```

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

```r
class(x)
```

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

```r
class('UCSD')
```

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

Object oriented programming makes languages flexible and powerful:

- You can create custom functions and objects for your needs
- Other people can create great packages for everyone to use
- Many errors come from using the wrong data type and a lot of programming in R is getting data into the right format and type to work with.

It is helpful to think in terms of object manipulation at a high level while programming in R, particularly at the beginning of tackling a new problem. Think about what objects you want to manipulate, what types they are, and how they fit together. Once you have the logic of your solution ready then you can write it in R.

2.3 Base-R vs. tidyverse

One last thing before we jump into data. Many things in R and other open source packages have competing standards. A lecture on a technique inevitably biases one standard over another. Right now among R users in this area, there are two families of functions: base-R and tidyverse. R instructors thus face a dilemma about which to teach primarily.[1]

In this math camp, we try our best to choose the one that is most useful to the modal task of social science researchers, and make use of the tidyverse functions in most applications. However, feel free to suggest changes to us or to the booklet.

Although you do not need to choose one over the other, for beginners it is confusing what is a tidyverse function and what is not. Many of the tidyverse *packages* are covered in this 2017 graphic below, and the cheat-sheets that other programmers have written: https://www.rstudio.com/resources/cheatsheets/

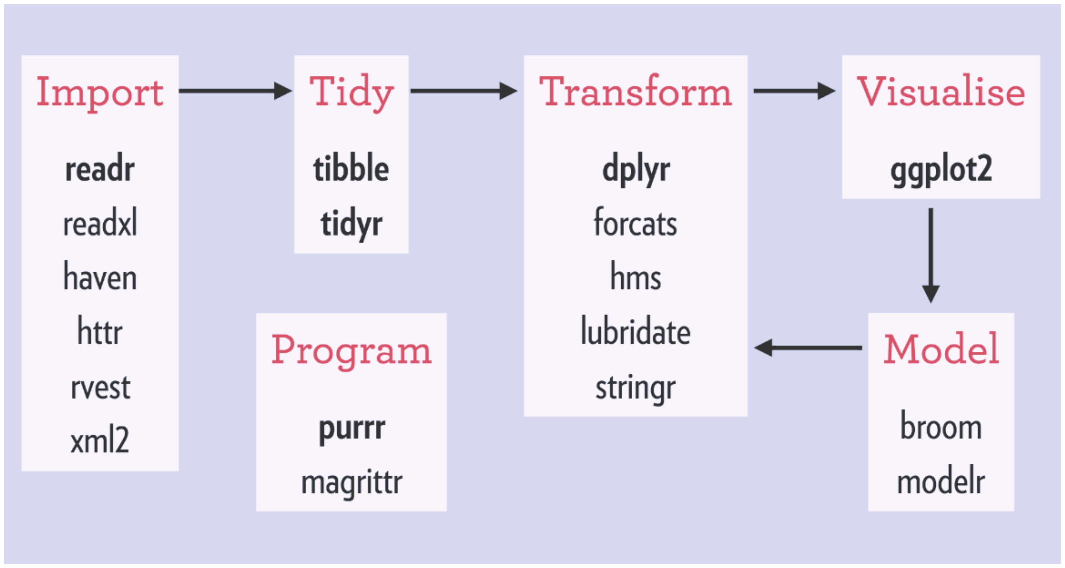


Figure 2.2: Names of Packages in the tidyverse Family

The following side-by-side comparison of commands for a particular function compares some tidyverse and non-tidyverse functions (which we refer to loosely as base-R). This list is not meant to be comprehensive and more to give you a quick rule of thumb.

Dataframe subsetting

In order to ...	in tidyverse:	in base-R:
Count each category	`count(df, var)`	`table(df$var)`
Filter rows by condition	`filter(df, var == "Female")`	`df[df$var == "Female", ]` or `subset(df, var == "Female")`
Extract columns	`select(df, var1, var2)`	`df[, c("var1", "var2")]`

[1]See for example this community discussion: https://community.rstudio.com/t/base-r-and-the-tidyverse/2965/17

In order to …	in tidyverse:	in base-R:
Extract a single column as a vector	`pull(df, var)`	`df[["var"]]` or `df[, "var"]`
Combine rows	`bind_rows()`	`rbind()`
Combine columns	`bind_cols()`	`cbind()`
Create a dataframe	`tibble(x = vec1, y = vec2)`	`data.frame(x = vec1, y = vec2)`
Turn a dataframe into a tidyverse dataframe	`tbl_df(df)`	

Remember that tidyverse applies to *dataframes* only, not vectors. For subsetting vectors, use the base-R functions with the square brackets.

Read data

Some non-tidyverse functions are not quite "base-R" but have similar relationships to tidyverse. For these, we recommend using the *tidyverse* functions as a general rule due to their common format, simplicity, and scalability.

In order to …	in tidyverse:	in base-R:
Read a Excel file	`read_excel()`	`read.xlsx()`
Read a csv	`read_csv()`	`read.csv()`
Read a Stata file	`read_dta()`	`read.dta()`
Substitute strings	`str_replace()`	`gsub()`
Return matching strings	`str_subset()`	`grep(., value = TRUE)`
Merge `data1` and `data2` on variables `x1` and `x2`	`left_join(data1, data2, by = c("x1", "x2"))`	`merge(data1, data2, by.x = "x1", by.y = "x2", all.x = TRUE)`

Visualization

Plotting by ggplot2 (from your tutorials) is also a tidyverse family.

In order to …	in tidyverse:	in base-R:
Make a scatter plot	`ggplot(data, aes(x, y)) + geom_point()`	`plot(data$x, data$y)`
Make a line plot	`ggplot(data, aes(x, y)) + geom_line()`	`plot(data$x, data$y, type = "l")`
Make a histogram	`ggplot(data, aes(x, y)) + geom_histogram()`	`hist(data$x, data$y)`
Make a barplot	See Section **??**	See Section **??**

2.4 A is for Athens

For our first dataset, let's try reading in a dataset on the Ancient Greek world. Political Theorists and Political Historians study the domestic systems, international wars, cultures and writing of this era to understand the first instance of democracy, the rise and overturning of tyranny, and the legacies of political institutions.

This POLIS dataset was generously provided by Professor Josiah Ober of Stanford University. This dataset includes information on city states in the Ancient Greek world, parts of it collected by careful work by historians and archaeologists. It is part of his recent books on Greece (Ober 2015), "The Rise and Fall of Classical Greece"[2] and Institutions in Ancient Athens (Ober 2010) , "Democracy and Knowledge: Innovation and Learning in Classical Athens."[3]

2.4.1 Locating the Data

What files do we have in the `data/input` folder?

```
## data/input/Nunn_Wantchekon_AER_2011.dta data/input/Nunn_Wantchekon_sample.dta
## data/input/acs2015_1percent.csv         data/input/gapminder_wide.Rds
## data/input/gapminder_wide.tab           data/input/german_credit.sav
## data/input/justices_court-median.csv    data/input/ober_2018.xlsx
## data/input/sample_mid.csv               data/input/sample_polity.csv
## data/input/upshot-siena-polls.csv      data/input/usc2010_001percent.Rds
## data/input/usc2010_001percent.csv
```

A typical file format is Microsoft Excel. Although this is not usually the best format for R because of its highly formatted structure as opposed to plain text (more on this in Section **??**(sec:wysiwyg)), recent packages have made this fairly easy.

2.4.2 Reading in Data

In Rstudio, a good way to start is to use the GUI and the Import tool. Once you click a file, an option to "Import Dataset" comes up. RStudio picks the right function for you, and you can copy that code, but it's important to eventually be able to write that code yourself.

For the first time using an outside package, you first need to install it.

```r
install.packages("readxl")
```

After that, you don't need to install it again. But you **do** need to load it each time.

```r
library(readxl)
```

The package `readxl` has a website: `https://readxl.tidyverse.org/`. Other packages are not as user-friendly, but they have a help page with a table of contents of all their functions.

[2] Ober, Josiah (2015). *The Rise and Fall of Classical Greece.* Princeton University Press.

[3] Ober, Josiah (2010). *Democracy and Knowledge: Innovation and Learning in Classical Athens.* Princeton University Press.