Data Science for Social Scientists: An applied course using IPUMS data

Developed by:
Daniel E. Ehrlich, IPUMS, University of Minnesota
Anna Tremblay, Dept of Soc, Anth, & CJ, Clemson University
Current Version compiled: 2023-01-04

Contents

Preface

An applied methods class for social scientists that uses real-world IPUMS data. This course is:

Open-source and customizable -
  All materials available on Github

Made with open-source tools -
  R, RStudio, bookdown

Driven by $^{\text{(nearly)}}$ open-source data -
  Harmonized across time and space: IPUMS

What is IPUMS

IPUMS started as a project to digitize the historical records of the US census. It has expanded to include 9 data collections, which are united in their methods and principles of making social science research easier. IPUMS data consists of individual-level census and survey data from more than 100 countries around the world. Notably:

- IPUMS **harmonizes** these data - ensuring consistently coded values across time and space.
- IPUMS provides harmonized **GIS Shapefiles** for most census and survey data.
- IPUMS provides extensive **metadata**, including:
  - Original questionnaire text.
  - Universe definition and comparability statements.
  - Alerts about notable changes in variable definition, universe, or coding schema

IPUMS data is free to use for education and research purposes. Researchers need only to register with an email address and brief project description. Nothing too formal - we're just trying to understand what kinds of questions researchers are interested in. For educators, we have additional resources to facilitate set up of classroom accounts - making it easy to get your students registered and share IPUMS data with them.

Why make this course

In a world where information and data are increasingly accessible, it is of utmost importance for individuals to understand data science and the interpretation of data. We believe that education should be easily accessible and teaching resources should be freely available to aid in this endeavor. While we (DEE) may be slightly biased, we think IPUMS is a fantastic resource for both **Education** and **Research**. Real-world example datasets provide the bulk of the content for this course, providing an applied context we hope students (and instructors) will find engaging. We also know many instructors may be teaching across multiple disciplines, in large departments, or be the only "data person" at their institution. We think IPUMS data is useful to virtually any social science field. We provide some example lessons, and encourage instructors to develop their own, using our `lesson_template.Rmd` to tailor this course to their subject or interest.

Getting Started

In order to use this textbook, you will need to:

- download and install RStudio
  - This link also contains instructions and links to download R from CRAN
  - Be sure you download the appropriate file for your Mac or PC
- Register for an with IPUMS account. We provide **limited example data**, but in order take full advantage of these exercises:
  - IPUMS registration for individuals
  - IPUMS registration for instructors

Course Description

This course is broken down into 3, 5-week units. Unit 1 focuses on familiarizing yourself with R and the IPUMS dataset. In Unit 2, each week will showcase a method/analysis using preselected variables. In class, students will walk through a given problem set and produce a lab report by the end of class. In Unit 3, students will work towards answering a research question that they pose, creating a research paper with literature review, data analysis, conclusion, and data outputs.

Course Aims

Provide students with relevant, hands on, methodological training in data literacy and visualization.

Learning Outcomes

After this course, students will be able to:

- Understand the depth of the IPUMS database and the variables it has to offer
- Compose R code to analyze the IPUMS data
- Produce visually pleasing data outputs in R
- Synthesize the information in a written report
- Present the analysis in a poster format for other students

Guiding Principles

- Phenomenon-based learning
  - try to start the class with a **question** or **problem**
  - *why* does the data look the way it does
  - structure class so students work towards solving the problem
- Relevant examples
  - Drawn from multiple disciplines (eg, economics, demography)
  - Can be added as modular examples/exercises

Syllabus - General

This syllabus is initially envisioned as 3 5-week sections. However, compilation and content are intended to be modular with templates for instructors to include their own specialties.

The basic structure of this course is:

Unit 1 (Weeks 1-5): Understanding and Testing Data

- Students use simple datasets bundled with the course or provided by the instructor.

- Simplified data to illustrate trends.
  - EG: plotting continuous variable (AGE); Table of categorical variable (SEX); Crosstabs

Unit 2 (Weeks 6-10): Finding Data and Asking Questions

- Students begin to analyze real world, IPUMS, datasets, provided by course/instructor.
- Students begin to model real world phenomena
  - EG: SEX ~ EDUATTAIN ; SEX ~ EDATTAIN + EMPSTAT
- Students learn to perform exploratory analysis, hypothesis testing, and statistical inference.
- Students learn to navigate IPUMS website,and find relevant data to thier research interest.

Unit 3 (Weeks 11-15): Discussing Data and Student Research

- Students develop a research question to be answered with IPUMS data.
  - Students are encouraged to fit it to their interests/major/discipline.
- Course time should be devoted to individual/small-group research.
- Instructor/class present on recent research.
  - Instructor models constructive / scholarly criticism.
  - Encourage students to critique published work - responsibly.

Syllabus - Detailed

Unit 1 Understanding and Testing Data

Students become gain familiarity and comfortability navigating RStudio, coding in R and performing simple data manipulation and visualization exercises. Datasets in this section consist of real-world (or synthetic) data, but the focus is on understanding data types (EG: using Age as a continuous variable; sex, education, employment as categorical; etc). Instructors should acknowledge these as **educational** datasets and make explicit trends found within these data are devoid of context, and must be taken with a (rather large) grain of salt, if at all.

By the end of Unit 1, students will be able to:

- Download R and RStudio

- Read data into R and
- Write (save) data out of R
- Summarize data visually
  - Using `base R`
  - Using `ggplot` (tidyverse)
- Summarize data in tables
  - Using base R
  - Using `gttable` / `tidyverse`
- Formally state and test assumptions of data
  - *EG:* t-test, anova, correlations, regression

By the end of Unit 1, students will understand

- Main types of data
  - *EG:* logical, numeric, character, etc
  - R specic vs general terms
- How to create and describe various data distributions
  - *EG:* normal, poisson, normal-skewed, etc
- Know which types statistical tests are appropriate for a given set of data.

Week 1: Intro to R, data types, data structures

Week 2: Plotting Data, Distributions

Week 3: Statisitcal testing of simple data sets

Week 4: Correlation and Relationships of simple data sets

Week 5: (TBD)

Unit 2 Finding Data and Asking Questions (Using IPUMS Data)

Here we demonstrate two **different** approaches to conducting research. Students become familiar writing up short lab reports detailing their findings. For Section **??**, we/instructor provides students with simple datasets from IPUMS (or other real-world data). Students will learn exploratory data analysis techniques and how to create lab reports to summarize key findings.

For unit **??**, students will learn to develop their own simple research questions or social-science hypotheses. They will seek out data to answer these questions, learning to navigate ipums.org, and create **data extracts**, as well as hypothesis-testing statistical methods. Again, lab reports to summarize findings.

0.0.0.1 Week 6: Intro to IPUMS

Week 7: Exploratory analysis

If you've just collected a survey, or other raw data, you may not know what you're looking for. This is perfectly ok but goes against *the scientific method* most people learned in grade school.

This unit begins by presenting data/distributions and asking students to begin interpreting the data . visual exploration is encouraged and basic of data manipulation are taught * *EG:* how to subset data, how to reshape data, how to re-code data, how to convert from one `data type` to another.

Example lab exercise:

Students given a data set (xls, csv, etc) * load data, perform manipulations, basic summaries + cross tabs + group means by a covariate * inspect data visually + *DESCRIBE* the distribution - is it normal? significant? * *FIND* aquestion in the spread of the data + how can you test this (maybe small group work) * write up/ present results + think on confounding factors / biases

Week 8: Hypothesis Testing

If, on the other hand you have an a pre-existing idea you want to test. We can follow the traditional *scientific method*. With a question in mind, the first question is: where to look. What better place than IPUMS!

Begin introducing navigation of web resources - mainly IPUMS international

Students should become comfortable working through lab exercises: * Define a question (or be presented with one) * Download variables from IPUMS (course downloads possible) * Perform a basic analysis (discussed in Unit 1) * Generate a **visual argument** for your analysis + Include explanation/interpretation/reflection on the question at hand, and the data used + Any obvious biases + Any obvious confounding factors

Week 9: Statistical Inference

Week 10: (TBD)

Unit 3 Discussing Data and Student Research

Students will select their own research question that can be answered with the IPUMS data set and will spend five weeks conducting a research project complete with data analysis, visualization, and interpretation.

In this section we encourage the instructor to provide ample time for independent student/small-group research. Some class time should be devoted to modeling healthy discussion and critique of methods. Students should learn to discuss not just *how* to answer a research question but *why* they are asking/answering it. What impact does the question/answers have. Is the question releveant/meaningful, and importantly, Is this research question perpetuating racist ideas.

We provide some examples here but encourage instructors (or students) to bring in recent journal/popular articles that do (or do not) apply data science methods well.

Week 11: Students develop research Question

Week 12: Students find relevant variables from IPUMS

Week 13: Students test and evaluate results

Week 14: Students prepare presentations of results

Week 15: Students present work (slides, poster, podium, etc)

DEV NOTES

TO DO

- **UPDATE TODO LIST**
- Make chapter 1 chapter 2
- Anna Adds chapter con data science intro exclusive of R/IPUMS
- discuss style
  - key terms section for each chapter?
  - key terms in **bold**
  - italics for *emphasis*
  - are we pro-hyphens, or are they pedantic?

MISC IDEAS

- Application forward
- Present research/ analysis/results FIRST, then explain the mathematical principals behind it
- daily/weekly "i'm stuck on…"
  - Students send in questions (night before class) and instructor spends 10-15 mins talking through (or collaboratively working through with class) solutions
  - Alternatively, once a month maybe a longer class covering "common problems asked this month" daily/weekly "recent research"
- pick out a recent article with good visualization (or bad) and spend 5-10 mins discussing what makes it good (or bad)
  - Encourage students to find articles for extra credit

Documentation

This function grabs any packages in your project and adds them to a local list that can be referenced using `R-pacakgename` * **NOTE** in practice, that needs to be wrapped in markdown syntax, eg: `[@R-bookdown]` * See help files for more info - might be able to create/add a `citation` file

Unit 1: The Basics

Lesson 0:

Lesson 0 files should contain a brief summary of the topics within each unit

Lesson 0 can also be used for a brainstorming space to sketch out ideas before creating `Unit#_Lesson#` files.

Lesson 1: What IS Data / Collecting / Visualizing Data

Lesson 2: Intro to R, data types, data structures

Lesson 3: Comparing Data

Lesson 4:

Lesson 5:

Unit-wide Glossary

Is this redundant?

Chapter 1

What *is* data?

1.1 POV:

In a social science class, your teacher tells you that the CDC reports average male height in the United States to be 69 inches or 5ft 9in. While browsing dating apps, you notice that nearly all the men report that they are 6ft or over. You wonder if this is a bias in reporting, or if the area where you live and attend college has significantly taller men. To test your theory, you want to collect data on height from individuals in your data science class to test if males are truly taller on campus than the country average.

Source: https://www.cdc.gov/nchs/fastats/body-measurements.htm**

1.1.1 ACTIVITY - collect data on height

If in-person * Create a histogram (x axis) on a blackboard/wall, have students place their heights with a post-it note + Start with one distribution for whole class + Repeat with separate distributions for M/F + *DEE: I want to find a way to acknowledge this dichotomy ignores non-binary individuals, and include some suggestions on how to discuss it.*

1.1.2 ACTIVITY - collect data on birth month

- Create a histogram (x axis) on a blackboard/wall, have students place their heights with a post-it note
  - Students place post-it notes on month of birth

IDEALLY: Have both a histogram of height AND histogram of birth month visible at the same time. Compare/contrast distributions of the two data sets.

1.2 Explore

Questions to consider: * *Why do we plot data* * *What does the* **distribution** *of post-it notes look like?* * *What can you infer from the distribution(s)* * *How does the distribution of height differ from birth month?*

1.2.1 Example Datasets

If you're working through this course on your own, or are unable to facilitate a classroom activity, see the companion R package, `ipumsED`, which includes example data sets for each lesson, as well as custom code and functions to facilitate learning data science with IPUMS data.

DEE: This could probably be stated at the begining of unit 1

1.2.2 ACTIVITY - Calculate by hand

In our hypothetical setup, we are interested in the **average** height. * *What does it mean to be* ***average*** * *Which height would you say is average? (eyeballing)* * *Which Birth Month would you say is average? - is there one?*

Write out steps for calculating **mean** - trivial as it may seem.

1.3 Explain

In the context of this example, we **collected** data on height and birth month for individuals in our class. Plotting our data allows us to **visualize** the data, making it easy to interpret.

1.4 Ellaborate

1.4.1 So what is data?

Data is defined as "facts and statistics collected together for reference or analysis."[1] As seen in Figure 1.1, there are two types of data: quantitative and qualitative. **Quantitative data** are able to be expressed in numerical format

[1]This is from the internet and needs to be our words

and are countable. These data are either discrete or continuous where **discrete data** uses numeric bins. For example, we use our age as discrete quantitative data, we round our age to the previous year (eg., 20, 21, 22). **Continuous data** does not use bins, but rather includes all of the fractions between two whole numbers. An example could be most physical measures like height, weight, the speed at which an individual runs.

Qualitative data describes characteristics or categories and can be broken down into two categories, nominal or ordinal. **Nominal data** has no inherent ordering but it can be categorized. Examples include country or origin, gender, hair color, race, etc. **Ordinal data** can both be categorized and ordered (e.g., first, second, and third place is a race).

> Going back to our hypothesis of male height on campus, heights are continuous, qualitative data. It is difficult for people to report their specific height and you assume that most individuals will report it rounded to the closest inch. This makes the data you will actually use, discrete quantitative data.

1.4.2 Collecting Data

The first step to answering a research question is to collect your data. Broadly, data comes in two forms, primary and secondary. (Fig 1.2) **Primary data** is data that is collected directly by the researcher. Surveys, observations, experimentation, questionnaires, and interviews are all examples of primary data. **Secondary data** is collected from published or unpublished literature. It is collected by different researchers and compiled for use by a second scientist. This type of data includes data found in published articles, books, journals, biographies, and government records like the US Census.

Once compiled, you now have a data set which is comprised of observations and variables. An **observation** is all of the measures taken for one person or item. A **variable** is what is being measured.

> The US CDC data is secondary, but you are collecting height data yourself in class as a comparison. The survey or questionnaire you use on your classmates is primary data. Each individual is an observation and the variable of interest is height.

1.4.3 POPULATIONS AND SAMPLING

Random Sampling: It is a sampling method in which all the items have an equal chance of being selected and the individuals who are selected are just like the ones who are not selected

Stratified Random Sampling: It is a process to gather data by separating the actual population into the distinct subset or strata, and then choosing simple random samples from each stratum Your research question is about the height of all males at your college, but recording height data for each individual would be very difficult and time consuming. You instead decide to use a sample of males in your data science class. This is a random sample as each male individual has an equally likely chance of being samples (that is, unless a prerequisite exists).

Sampling strategy can lead to **bias**

> If you had chosen a different sample, like the men's basketball team, your results would have been biased.

1.5 Evaluation

1.5.1 Review Questions

- What is one example of collecting/visualizing data from your own life
- Is height a **continuous** or **discrete** variable and Why?

1.5.2 Exercises

Brainstorm 3 topics/questions that are of interest to you personally, or academically. * What **variable(s)** will you need to collect to study this phenomenon? * Describe these variable(s), are they qualitative or quantitative? Continuous or ordinal?

Glossary

Chapter 2

Intro to R, data types, data structures

In the previous lesson, we began thinking about **data**, how to talk about it, and how to **visualize** it. We also talked about one type of average, the **mean.**

2.1 Engage

What do you think the following code does?

```
my_data <- read_excel("//filepath/directory/filename.xlsx")
```

Hint: There are 4 "things" in the above code: `my_data`, `<-`, `read_excel()`, `//filepath/directory/filename.xlsx`

2.1.1 R vs RStudio

- If `R` is the engine, then `RStudio` is the car.
- If `R` is the text, Rstudio is the text-editor.
- `R` is the **programming language**, `Rstudio` is the **Integrated Development Environment (IDE)**.
  - *What do you think some differences are between R and RStudio?*

`RStudio` is a program/app just like Google Chrome or Microsoft Word. Each of these programs provide a **Graphical User Interface (GUI)**, a pretty way for a user to use a mouse and keyboard to do *something.*

An **IDE** is a special kind of program/app that provides MANY tools for writing and running code.

2.1.2 Orientation to RStudio

SUGGESTION: Instructor live demos interacting with RStudio while students follow along on computers

When you first open RStudio, you'll see 3 **panes**, all of which will look fairly empty at the moment.

If you're using the default layout, you should see: * On the left, the `Console` **pane** * On the Top-right, the `Environment` **pane** * On the Bottom-right, the `Files` **pane**

A keen eye will also notice that each of these **panes** contains multiple **tabs**. We will go over the uses of many of these **tabs**, but for now let's start with `Console.`

The `Console` is where you input `R code`, run it, and see the results.

At it's simplest `RStudio` is a calculator. Try typing `4 + 4` into the `Console`, then press the `[Enter]` key to run the code. Immediately, `R` prints the result as we see here:

```
4 + 4
```

```
## [1] 8
```

The `Console` serves as a running log of all your operations for your current session, but it can be helpful to temporarily save your results as `R objects`. We do this using the **assignment operator** we saw earlier: `<-`

```
answer <- 4+4
```

By default, `Console` only prints results if they have no where else to go. If they're stored as an `Robject`, no result is printed, but you should now see `answer` listed in the `Environment` **pane**.

2.2 Explore - Interacting with R objects

2.2.1 Load the Data

```
dir_path <- file.path("inst","unit1_data")
survey_path <- file.path(dir_path, "data_template.xlsx")

data <- readxl::read_excel(survey_path)
```

What is `data`?? Below we call the `class()` function on `data` and see that it has 3 classes: `tbl_df` , `tbl` , `data.frame`

The first two classes, `tbl_df, tbl` indicate it is a special kind of table, in the `tibble` format. In general, you can interact with these like a `matrix` or `data.frame` but they have additional features.

```
class(data)
```

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

We can call `colnames()` on data, like a regular `data.frame` or `matrix`. Or we can take advantage of the `tibble` structure and use the `glimpse()` function which provides a succinct summary of your data.

```
colnames(data)
```

```
## [1] "individual"    "Birth_Month"   "Height_inches"
```

```
tibble::glimpse(data)
```

```
## Rows: 32
## Columns: 3
## $ individual    <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1~
## $ Birth_Month   <chr> "January", "September", "March", "April", "April", "Octo~
## $ Height_inches <dbl> 70, 64, 72, 61, 55, 65, 72, 75, 69, 75, 76, 70, 70, 69, ~
```

2.2.2 Inspect the Data

What is `data`?? Below we call the `class()` function on `data` and see that it has 3 classes: `tbl_df` , `tbl` , `data.frame`

The first two classes, `tbl_df, tbl` indicate it is a special kind of table, in the `tibble` format. In general, you can interact with these like a `matrix` or `data.frame` but they have additional features.

```
class(data)
```

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

We can call `colnames()` on data, like a regular `data.frame` or `matrix`. Or we can take advantage of the `tibble` structure and use the `glimpse()` function which provides a succinct summary of your data.

```
colnames(data)
```

```
## [1] "individual"    "Birth_Month"   "Height_inches"
```

```
tibble::glimpse(data)
```

```
## Rows: 32
## Columns: 3
## $ individual    <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 1~
## $ Birth_Month   <chr> "January", "September", "March", "April", "April", "Octo~
## $ Height_inches <dbl> 70, 64, 72, 61, 55, 65, 72, 75, 69, 75, 76, 70, 70, 69, ~
```

Summarize Data

Continuous Data

For continuous data, we often want to summarize our data by describing the **mean, median, and/or range**. **Mean** and **median** describe the *central tendency* of the data, while **range** describes the full extant of the data, as seen below.

NOTE: if `NA` are present in the data, be sure to use the `na.rm=TRUE` flag for these operations.

```
mean(data$Height_inches, na.rm = T)
```

```
## [1] 66.5625
```

```
median(data$Height_inches, na.rm = T)
```

```
## [1] 66.5
```

```
range(data$Height_inches, na.rm = T)
```

```
## [1] 55 76
```

All in summary()

Mean, **median**, and **range** will all be reported by calling `summary()` on a `numeric vector`, such as `Height_inches`. In addition, the lower and upper quartiles will be reported, along with the number of `NA` responses.

NOTE: `summary()` does NOT require special handling for `NA` values, in fact - it expects them!

```
summary(data$Height_inches)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   55.00   64.00   66.50   66.56   70.00   76.00
```

2.3 Mode

You're probably familiar with **mean** and **median** being talked about with a third term, **mode**. The **mode** is the most commonly occuring value in a dataset. It's often important to know the **modal response** of survey data. While a commonly reported metric(??), there is no `mode()` function included in `base r`…

so we'll just have to create our own!

2.3.1 Mode Code

One common measure of data reported is the mode, or most frequently occuring value. For whatever reason, this is not a default function in R, but we can easily write our own function like so:

```
my_mode <- function(x){
  tt <- table(x) ## find frequencies
  tt <- tt[order(tt, decreasing = TRUE)] ## resort based on freq

  ## check number of modes
  max <- max(tt)
  n_max <- sum(tt==max)
  if(n_max > 1 ){
    warning("More than one mode detected")
    return(tt[tt==max])
  } else {
    ## return only the first value
    return(tt[1]) ## return whatever the highrst frequency is
  }


}
```

2.3.2 Mode Results

Now that we've created out own `function`, it's easy to find the **mode**

```
my_mode(data$Height_inches)
```

```
## Warning in my_mode(data$Height_inches): More than one mode detected
```

```
## x
## 64 65 69 70
##  4  4  4  4
```

2.4 Visualizing Data

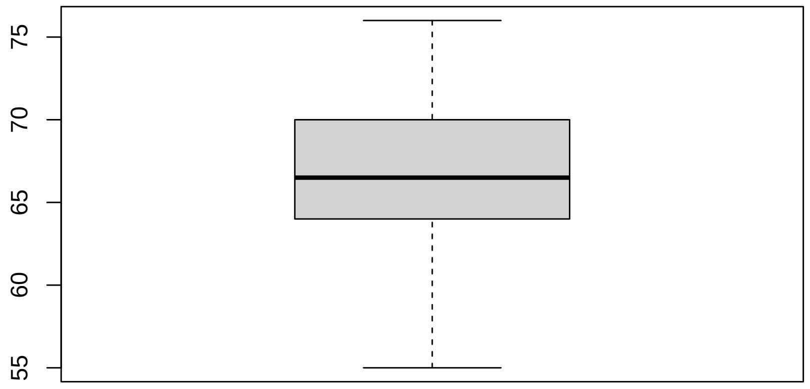
The above summaries describe data with numbers, but we can also describe data visually.

2.4.1 Continuous Data - Boxplots

Univariate continuous data, like height, can be visualized using a box and whisker plot, which shows many of the components of summary:

- the **median** is the black bar in the middle
- the **quartiles** (25th and 75th percentiles) are represented by the extents of the boxes
- The **range** is shown by the whiskers, with outliers shown indvidually, if needed.

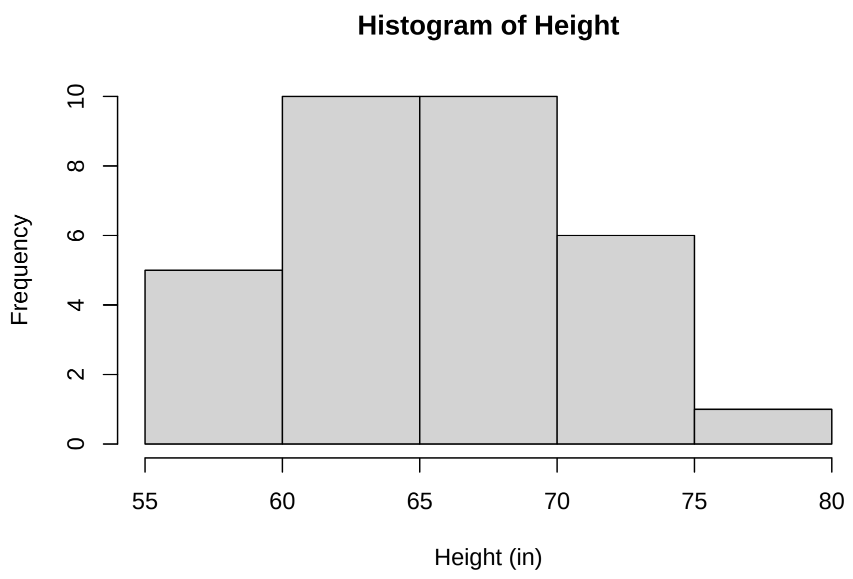
```
boxplot(data$Height_inches)
```



2.4.2 Continuous Data - Histograms

Continuous data, can also be broken into **bins** and plotted as a **histogram**. The `hist()` function will attempt to find the optimum number of bins for you, but you can specify a different number with the `breaks` argument.

```
hist(data$Height_inches, main = "Histogram of Height", xlab = "Height (in)")
```



2.4.3 Categorical Data

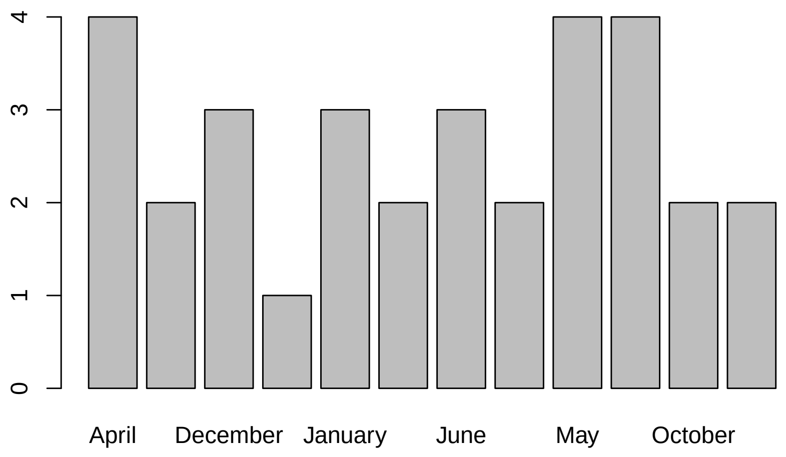
Categorical data is already in discrete units. In general with categorical data, we want to count the **frequency** of unique values. There are many ways to do this, but one of the easiest is the `table()` function. Saving the results of the table to an object, `birth_freq`, allows you to save and print the results at any time.

```
birth_freq <- table(data$Birth_Month)

birth_freq
```

```
## 
##     April    August  December  February   January      July      June     March 
##         4         2         3         1         3         2         3         2 
##       May  November   October September 
##         4         4         2         2
```

We can also visualize our tabulated results using a **barplot** as below.

```
barplot(birth_freq)
```



Glossary

Chapter 3

Comparing Data

NEEDS A LOT OF WORK

3.1 Data Distributions

3.1.1 Normal Distributions

First we'll generate a normal distribution with the `rnorm()` function. This takes 3 arguments: `n, mean, sd`, which you can see filled in below. While we could print out a list of all these values, it's not easy to *understand* a list of numbers

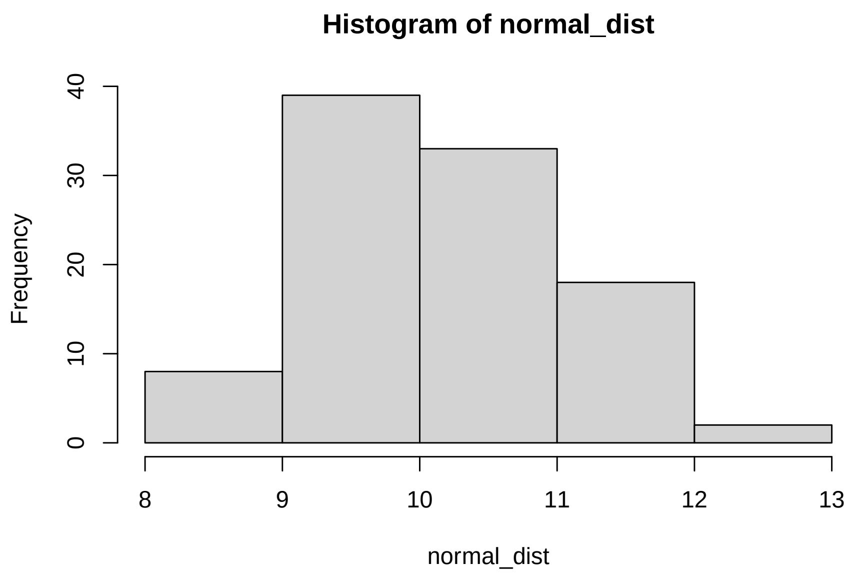
```
normal_dist <- rnorm(n = 100, ## 100 samples
                     mean = 10, ## with a mean of 10
                     sd = 1 ## and a standard deviation of 1
                     )


normal_dist
```

```
##   [1] 11.461743  9.228759 10.344360  9.011577 10.213482 10.412601  9.794298
##   [8] 10.493866  9.504083 11.064996  9.699334  9.528765  9.695063 10.451686
##  [15] 10.087802 11.458674  9.130766 11.169342 10.501660 10.984828 11.153812
##  [22]  9.105005  8.527781  9.019875 10.992873  9.472671  9.630453 11.930379
##  [29]  9.709972  8.915032  9.977849  9.884639 11.264260  8.107857 10.610282
##  [36]  9.271024 11.340466  9.056360  9.240962 10.841711  9.398378 10.762601
##  [43] 10.920869 10.909921 12.515859  9.062207 10.473640  9.640756 10.307330
##  [50]  9.751568  9.465202  9.822931 10.427316  8.177950 10.831855  9.578186
##  [57]  9.241150 11.090053 10.497670  8.966486  8.757531  8.188186 10.269957
##  [64] 12.410783  9.922728 11.798599  9.000495  9.854824  9.246429 11.089729
##  [71] 10.751389 10.548314 11.734746  9.533314  9.581124 10.169512 10.257509
##  [78]  9.636182  8.602800 10.572997  9.692793  9.875417 11.153376 10.208966
##  [85] 11.528380 10.993645  9.387126 10.528465 11.308611 10.159791 10.715990
##  [92] 11.469583  9.847889  9.435087  9.826541 10.593494 11.558613 11.376639
##  [99] 10.062645 10.289308
```

Another better way to look at data would be to **visualize** or **plot** it. One way to to that is with a **histogram**, which groups **continuous values** into **bins**, then plots the **frequency** for each bin.

In R, we use the `hist()` function to plot a histogram of data. We can (try to) control the number of bins with the `breaks` argument, but note that it doesn't always match up. The `hist()` function will adjust based on the distribution of the data.

```
hist(normal_dist,breaks = 5)
```



Another way to visualize this would be with a d

3.1.2 What *is* normal?

3.1.2.1 Quantitative summaries

5num summary * Min, 25th percentile, median, 75th percentile, Max

```
tab_normal_dist <- summary(normal_dist)
```

We can print the table in R by calling its name.

```
tab_normal_dist
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   8.108   9.496  10.165  10.161  10.859  12.516
```

Mean, standard deviation

3.1.2.2 Meaningful Comparisons

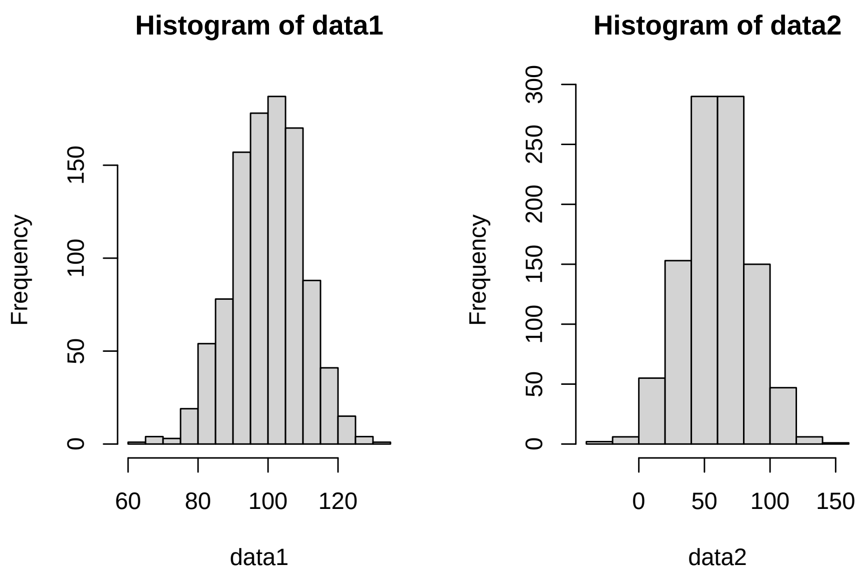
How to compare apples to oranges? Standardize the units / standardize the data

```
data1 <- rnorm(n=1000,
              mean = 100,
              sd = 10)

data2 <- rnorm(n=1000,
               mean = 60,
               sd = 25)
```

Are these the same distribution?

Any issues??

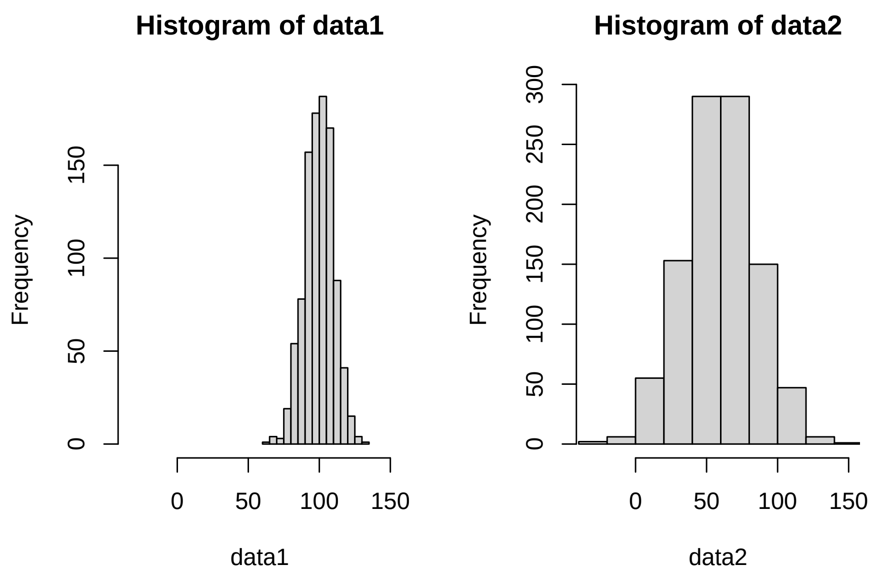
```
layout(matrix(1:2, ncol = 2))
hist(data1)
hist(data2)
```



```r
total_range <- range(data1, data2)
```

Are they the same?

```r
layout(matrix(1:2, ncol = 2))
hist(data1, xlim = total_range)
hist(data2, xlim = total_range)
```



Numerically / tabularly

Often times its important to tables of **summary statistics**

```
norm_comp_tab <- rbind(summary(data1),
                       summary(data2))

norm_comp_tab
```

```
##            Min.  1st Qu.    Median     Mean   3rd Qu.     Max.
## [1,]  61.97265 93.25935 100.31955 99.83822 106.85430 133.9527
## [2,] -34.25198 42.98023  59.75894 59.52261  76.50663 150.4806
```

Making the table a little nicer. Also an example of **conditional programming**.

```
rownames(norm_comp_tab) ## they're null
```

```
## NULL
```

```
if(is.null(rownames(norm_comp_tab))){
  rownames(norm_comp_tab) <- c("data1", "data2")
}
```

When working with **Rmarkdown** we can take advantage of `knitr` and `pandoc` to nice looking tables even easier.

```
knitr::kable(norm_comp_tab)
```

	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
data1	61.97265	93.25935	100.31955	99.83822	106.85430	133.9527
data2	-34.25198	42.98023	59.75894	59.52261	76.50663	150.4806

How transform the data

Simple transformation (multiply all values by 100) * to convert units * other examples?

Complex transformations * log-transformation (*DEE: not a fan*) * z-scores (*DEE: a better option*)

Why transform the data? * Real world applications? * Is it always appropriate to transform data?

3.1.3 Skews

What to do if the data are **not** normal?

3.2 Statisitcal testing of simple data sets

3.2.1 t-tests, ANOVA, chi2

3.3 Relationships between variables in simple data sets

3.3.1 Correlation, Linear Regression

3.3.1.1 Simple LM

3.3.1.2 Complex LM

3.3.2 Genearlized Linear Model

For now, I have 3 main chapters for each of the main sections: * Basics of data science / R **??** * Applications/critiques using IPUMS data **??** * Student-driven projects **??**

Each of these **Chapters** contains multiple sections. We'll likely want to break these sections out into their own `.Rmd` files as they get fleshed out. For now, I'll try to keep the abundance of files limited.

NOTE: As these actually get filled out, we will probably want to insert different `parts` to the book (EG, the content of Unit 1 is covered in `Part I`). * Declare parts with `# (PART) Part I {-}` immediately before the first chapter `#` it contains.

Topics to include: * What is data? * Everything can be data * How do we interpret data * Tables * Plots * Univariate distributions * What can they tell us * Multi-modality in distributions * Categorical vs continuous data * Don't need to get ahead of this yet * Add in a grouping category - multi state/multi-national dataset * Ttest / anova

Type of Data: Age distributions Specifically generate a dataset with old/young folks over-represented to highlight a bimodal distribution

Start with single state/country Add a second state/country to demo ttest Add more to demo anova

Alternatively, income by education level - may be more interesting/relevant to college students (or depressing)

3.4 Intro to R/RStudio

3.5 Reading Data / Distributions

3.5.1 What *is* a normal distribution

3.5.1.1 How normal is it?

show increasingly unclear examples of normal vs not

introduce tests of normality

3.5.1.2 Measuring normality - single sample

reinforce [concept of statistical] **normality**

is a value from a sample? - one way ttest something about tails

3.5.1.3 comparing normality - two saples

standard / two-way t test

3.5.1.4 comparing more than two - ANOVA

Glossary

Data Quantitative Qualitative Discrete Continuous Nominal Ordinal

Unit 2: IPUMS

Lesson 6 Introduction to IPUMS

Some text to break up the sub-section headers

Intro to IPUMS website

background on ipums

navigating website

Find certain (very common) variables to answer (common) social science questions.

Lesson 7 Exploratory analysis

If you've just collected a survey, or other raw data, you may not know what you're looking for. This is perfectly ok but goes against *the scientific method* most people learned in grade school (More on that to follow(***include_link***)).

This unit begins by presenting data/distributions and asking students to begin interpreting the data . visual exploration is encouraged and basic of data manipulation are taught * *EG:* how to subset data, how to reshape data, how to recode data, how to convert from one `data type` to another.

Example lab exercise:

Students given a data set (xls, csv, etc) * load data, perform manipulations, basic summaries + cross tabs + group means by a covariate * inspect data visually + *DESCRIBE* the distribution - is it normal? significant? * *FIND* aquestion in the spread of the data + how can you test this (maybe small group work) * write up/ present results + think on confounding factors / biases

Advanced Exploration - Change Over Time

Here we demonstrate an approach to looking at how Family Structure (inferred from household relationships) has changed over time.

Setup / Load Data

Install/update R packages

```
install.packages("ipumsr")
install.packages("tidyverse")
```

Data extract created online using the datacart system.

```
library(ipumsr)
library(dplyr)


ddi <- read_ipums_ddi("Data/ipumsi_00005.xml")
data <- read_ipums_micro(ddi)
```

Inspect the Data Using `haven` labeled values.

```
data$RELATE[1:100]
class(data$RELATE)

data %>% count(RELATE)
data %>% count(SEX)
```

What were those codes ??

```
## need to convert this to an image or something similar; kable table?
ipums_view(ddi)
```

Visualize A simple plot

```
plot(AGE ~ YEAR, data = data)
```

A fancier plot

```
plot(AGE~YEAR, data = data, type = "n", main = "Age by Sex, over Time, CO")
points(data$YEAR[data$SEX==1]-1, data$AGE[data$SEX==1], pch = 16, col = hsv(.6,.6,.8,.2))

points(data$YEAR[data$SEX==2]+1, data$AGE[data$SEX==2], pch = 16, col = hsv(1,.6,.8,.2))

abline(lm(AGE~YEAR, data = data), col = "green")
```

Asking (logical) questions Here we demonstrate how setting up logical questions can be used to easily filter/subset data.

```
age_test <- data$AGE > 18

class(age_test)

age_test
```

Logical vectors are stored as `TRUE` or `FALSE`, but can also be evaluated numerically as `1` or `0` respectively. We can therefore `sum()` the number of `TRUE` values and divide by total rows for a proportion.

```
sum(age_test)/nrow(data)
```

HH vs persons A unique characteristic of census and some survey data is the nested-structure with individuals being grouped into households. Often times it is necessary to choose to work at the hh or person level, and data must be appropriately manipulated to fit that case.

```
hh_total <- length(unique(data$SERIAL))
hh_total
ipums_view(ddi)
```

Nuclear Family

First we look at a nuclear family, comprising only parents and their immediate children.

```
library(ipumsr)
library(dplyr)

ddi <- read_ipums_ddi("/pkg/ipums/personal/ehrli097/AABA_2022/Data/ipumsi_00005.xml")
all_data <- read_ipums_micro(ddi)
```

```
census_years <- c(1860, 1870, 1880, 1900, 1910, 1960, 1970, 1980, 1990, 2000, 2010)

## subset census only
d2 <- all_data %>% filter(YEAR %in% census_years)

## make a household dataframe
hhs <- d2 %>% distinct(YEAR, SERIAL, .keep_all = TRUE) %>% select(YEAR,SERIAL,GEO1_US)

hhs %>% View()
```

```
hhs <- d2 %>% filter(RELATE ==4) %>%


  distinct(YEAR, SERIAL) %>% mutate(extended_test=TRUE) %>% right_join(hhs, by = c("YEA









hhs <- d2 %>% filter(!RELATE %in% c(1, 2, 3) |
                  (RELATE == 3 &
                     MARST %in% c(2, 3, 4))
                ) %>%




  distinct(YEAR, SERIAL) %>% mutate(nuclear_test = FALSE) %>% right_join(hhs, by = c("Y



table(hhs$extended_test,hhs$nuclear_test)
```

```


  hhs <- d2 %>% filter(RELATED %in% c(4200, 4210, 4211, 4220, 4500, 4510, 4600)) %>% di


  hhs <- d2 %>% filter(RELATED %in% c(4100, 4110, 4120, 4130, 4300, 4301, 4302)) %>% di
  res_tabs <- list(
    "nuclear_test" = hhs %>% group_by(YEAR, nuclear_test,GEO1_US) %>% summarize(.groups="drop",n
  "extended_test" = hhs %>% group_by(YEAR, extended_test, GEO1_US) %>% summarize(.groups="drop",r
  "parent_test" = hhs %>% group_by(YEAR, parent_test, GEO1_US) %>% summarize(.groups="drop",n = r
  "children_test" = hhs %>% group_by(YEAR, children_test, GEO1_US) %>% summarize(.groups="drop",r
  )






collapsed_results <- res_tabs %>% purrr::map(function(x){
  x <- x %>% group_by(across(names(x)[1:3])) %>% summarize(.groups="drop",n = sum(n))

})


collapsed_results <- lapply(collapsed_results, function(x){
  colnames(x)[2] <- "test"
  colnames(x)[3] <- "state"
  return(x)
})

combined <- collapsed_results %>% purrr::reduce(full_join, by = c("YEAR", "test", "state"))



colnames(combined) <- c("YEAR","test", "state", "n_nuclear", "n_extended", "n_parent", "n_childre

combined[is.na(combined)] <- 0


to_plot <- combined %>% group_by(YEAR, state) %>% mutate(n_tot = sum(n_nuclear)) %>% ungroup() %>
```

Tabulate results

```
to_plot <- to_plot %>% filter(test==TRUE)



plot(to_plot$YEAR, to_plot$pct$n_nuclear, col = hsv(.4, .6,.8), pch = 16, ylim =c(0,1), xlab = "'
```

Visualize Nuclear Families

Extended Family

Next we look at hhs with extended families present. IE, any that contain more relationships than just Parent/Child/Sibling (between children only)

```
to_plot <- to_plot %>% filter(test==TRUE)


glm_hist <- glm(pct$n_extended ~ YEAR, data = to_plot[to_plot$YEAR < 1950,], family = q



glm_hist_x <- seq(from=1860, to = 1910, length.out = 100)
glm_hist_y <- predict(glm_hist, list(YEAR = glm_hist_x), type = "response")

glm_mod <- glm(pct$n_extended ~ YEAR, data = to_plot[to_plot$YEAR> 1950,], family = qua

glm_mod_x <- seq(from = 1960, to = 2010, length.out = 100)
glm_mod_y <- predict(glm_mod, list(YEAR = glm_mod_x), type = "response")

mods <- list("hist"=list(),
             "mod" = list()
             )
mods_plots <- list("hist"=list(),
                   "mod" =list()
                   )

for(i in names(to_plot$pct)){

  hist_x <- to_plot$YEAR[to_plot$YEAR < 1950]
mod_x <- to_plot$YEAR[to_plot$YEAR > 1950]

  mods$hist[[i]] <- lm(pct[[i]] ~ YEAR, data = to_plot[to_plot$YEAR < 1950,])

  mods_plots$hist[[i]] <-
    data.frame("x" = hist_x,
               "y" = predict(mods$hist[[i]],
                             list(YEAR =hist_x),
                             type = "response")
               )
  mods$mod[[i]] <- lm(pct[[i]] ~ YEAR, data = to_plot[to_plot$YEAR > 1950,])


  mods_plots$mod[[i]] <-
    data.frame("x" = mod_x,
               "y" = predict(mods$mod[[i]],
                             list(YEAR =mod_x),
                             type = "response")
               )
}
```

Gernerate models

```
plot(to_plot$YEAR, to_plot$pct$n_extended, col = hsv(.95, .6,.8), pch = 16, ylim =c(0,.25), bg =


lines(glm_hist_x,glm_hist_y, col = hsv(.95, .3, 1), lwd = 2)
lines(glm_mod_x, glm_mod_y, col = hsv(.95, .3, 1), lwd = 2, lty = 2)




points(to_plot$YEAR,
       to_plot$pct$n_extended,
       pch = 23,
       bg = hsv(.95,.6,.8))
```

Visualize

Even more DETAIL - maybe remove

```
ipums_view(ddi)
```

```
  hhs <- d2 %>% filter(RELATED %in% c(4200, 4210, 4211, 4220, 4500, 4510, 4600)) %>%

  distinct(YEAR, SERIAL) %>% mutate(parent_test=TRUE) %>% right_join(hhs, by = c("YEAR", "SERIAL'

  hhs <- d2 %>% filter(RELATED %in% c(4100, 4110, 4120, 4130, 4300, 4301, 4302)) %>% distinct(YEA
plot(to_plot$YEAR, to_plot$pct$n_extended, col = hsv(.95, .6,.8), pch = 16, ylim =c(0,


lines(glm_hist_x,glm_hist_y, col = hsv(.95, .3, 1), lwd = 2)
lines(glm_mod_x, glm_mod_y, col = hsv(.95, .3, 1), lwd = 2, lty = 2)


lines(mods_plots$hist$n_parent,col = hsv(.8, .3,1), lwd = 2)

lines(mods_plots$mod$n_parent, col = hsv(.8, .3,1), lwd = 2, lty = 2)

points(to_plot$YEAR,
       to_plot$pct$n_parent,
       pch = 23,
       bg = hsv(.8, .6,.8))


points(to_plot$YEAR,
       to_plot$pct$n_extended,
       pch = 23,
       bg = hsv(.95,.6,.8))
```

Parents Supporting Parents

```
plot(to_plot$YEAR, to_plot$pct$n_extended, col = hsv(.95, .6,.8), pch = 16, ylim =c(0,


lines(glm_hist_x,glm_hist_y, col = hsv(.95, .3, 1), lwd = 2)
lines(glm_mod_x, glm_mod_y, col = hsv(.95, .3, 1), lwd = 2, lty = 2)



lines(mods_plots$hist$n_children, col = hsv(.55,.3,1), lwd = 2)


lines(mods_plots$mod$n_children, col = hsv(.55,.3,1), lwd = 2, lty = 2)


points(to_plot$YEAR,
       to_plot$pct$n_children,
       pch = 23,
       bg = hsv(.55,.6,.8))

points(to_plot$YEAR,
       to_plot$pct$n_extended,
       pch = 23,
       bg = hsv(.95,.6,.8))
```

Parents Supporting (extended) children

Lesosn 8: Hypothesis Testing

If, on the other hand you have an a pre-existing idea you want to test. We can follow the traditional *scientific method*. With a question in mind, the first question is: where to look. What better place than IPUMS!

Begin introducing navigation of web resources - mainly IPUMS international

Students should become comfortable working through lab exercises: * Define a question (or be presented with one) * Download variables from IPUMS (course downloads possible) * Perform a basic analysis (discussed in Unit 1) * Generate a **visual argument** for your analysis + Include explanation/interpretation/reflection on the question at hand, and the data used + Any obvious biases + Any obvious confounding factors

Lesson 9: Statistical Inference

Lesson 10: (TBD)

We describe our methods in this chapter.

Math can be added in body using usual syntax as follows. This may be useful, particularly for explaining the math side of things.

Unit 3: Independent Research

Students will select their own research question that can be answered with the IPUMS data set and will spend five weeks conducting a research project complete with data analysis, visualization, and interpretation.

In this section we encourage the instructor to provide ample time for independent student/small-group research. Some class time should be devoted to modeling healthy discussion and critique of methods. Students should learn to discuss not just *how* to answer a research question but *why* they are asking/answering it. What impact does the question/answers have. Is the question releveant/meaningful, and importantly, Is this research question perpetuating racist ideas.

We provide some examples here but encourage instructors (or students) to bring in recent journal/popular articles that do (or do not) apply data science methods well.

Lesson 11: Students develop research Question

Lesson 12: Students find relevant variables from IPUMS

Lesson 13: Students test and evaluate results

Lesson 14: Students prepare presentations of results

Lesson 15: Students present work (slides, poster, podium, etc)

By this point, students should be familiar with basic concepts from Chapter Unit 1. These include:

- Basic Coding
  - read/write data in/out of R
  - basic manipulations
- Theoretical Basis
  - looking at data distributions
  - formal assessment of distributions

Students will also be familiar with how these concepts are applied from Chapter Unit 2. Hopefully students will be able to:

- Come up with a social science question they are interested in
  - Critically think about target variable(s) of interest. Any *a priori* covariates? confounders?
  - Acquire relevant data from IPUMS
  - Analyze, Summarize, Visualize Data
    * scope and complexity at student/teach discretion
  - Present research to class
    * **potentially** critically discuss/evaluate each others work.
    * **science is collaborative** everyone should be out to do their best work and represent the data as best we can. We all have conscious and unconscious biases, and the best way to confront them is share and receive (respectful) feedback.

During this Unit, we suggest giving ample class time for independent student research, peer-to-peer collaboration, and basic R/stats troubleshooting. This would also be a great time to model how to give respectful criticism by discussing recent research papers. * We could maybe come up with 1-2 seed examples, with a few talking points

3.5.2 Example one

3.5.3 Example two

Chapter 4

Example RMD code

For now, this chapter is a bit of a placeholder. I'm not sure what/how the `references.Rmd` file actually fits in to the code/construction (it looks automatic) so I want to keep that in place and need a section to note that.

I also want a more centralized reference point to put any example code I find helpful while working in R/bookdown. This section could get really unrully really fast, but oh well.

4.1 Core

`index.Rmd` is required and treated as file `00`. Chapters *should* be numbered for ease of sorting but custom orders are possible by specifying filenames somewhere **in this file**

Remember each Rmd file contains one and only one chapter, and a chapter is defined by the first-level heading `#`. + **IE** beyond the YAML header this file functions as a normal chapter since it starts with a top level header. + Note that `index.Rmd` has its own YMAL in addition to the various .yml files…not sure exactly how these relate.

Reference a figure by its code chunk label with the `fig:` prefix, e.g., see Figure @ref(fig:norm_dist_plot). Similarly, you can reference tables generated from `knitr::kable()`, e.g., see Table @ref(tab:norm_summary_tab). * Again, this prints an auto-generated numeral * also leaving this in the context of the plots in Chapter **??**

You can write citations, too. See `knitr::write_bib()` for more on this. Quick example from demo/index (may not work without write_bib() though): we are using the **bookdown** package (Xie, 2022) in this sample book, which was built on top of R Markdown and **knitr** (Xie, 2015). * If included, “Refernces” section gets added to each chapter. * Not exactly sure where

Embed html renders (EG, fancy tables (IPUMS_var_desc), or any shiny app) with `webshot` R package and `phantomJS`.

```
install.packages("webshot")
webshot::install_phantomjs()
```

Embed figures from a folder.

For this, it’s usually best to use a code-chunk and `knitr`. There are a number of graphical paramerters you can set (or ignore) `out.width` will scale your image accordingly - irrespective of unit/display `fig.align` should be “left”, “right”, or “center” `fig.cap` allows you to provide “mouse over” captions for the image. `echo=FALSE` is important if you ONLY want the image (IE the result of the code). If you want the code itself to show, (IE, or echo) set `echo=TRUE`.

Figure 4.1: the ipums logo

4.2 Tips

***Autonumber sections** Note the `{-}` used to indicate “do not number this section” eg: preface.

LABEL EVERYTHING you’ll likely want to reference it later * code chunks that produce figures can be referenced via `@\ref(fig:[LABEL])`

You can label chapter and section titles using `{#label}` after them, e.g., we can reference Chapter **??**. If you do not manually label them, there will be automatic labels anyway, * No idea how the automatic references work, so always be sure to declare them. * **NOTE** these display as the relevant Chapter `numeral`.

4.3 Syntax

italics or *italics* (can handle spaces) **bold** `code` *equations*

4.3.1 Math

Randal Pruim features an extensive list of common math expression on their github page. Here are some quick notes:

In-line equations can be written within `$` and will be displayed right there: $a^2 + b^2 = c^2$. In contrast, you can also add equation chunks by using `$$`

This can be coded in-line,

$$\sum_{n=1}^{10} n^2$$

, but will result in a page break.

Alternatively, a more "classic" equation chunk:

$$ Plain text doesnt get spaces

how

very

odd

$$

4.3.1.1 more math example

p is unknown but expected to be around 1/3. Standard error will be approximated

$$SE = \sqrt(\frac{p(1-p)}{n}) \approx \sqrt{\frac{1/3(1-1/3)}{300}} = 0.027$$

You can also use math in footnotes like this[1]. Footnotes are helpful because they re-link to where you left off.

We will approximate standard error to 0.027[2]

The `longnote` footnote seems particularly useful.

To compile this example to PDF, you need XeLaTeX. You are recommended to install TinyTeX (which includes XeLaTeX): https://yihui.name/tinytex/.

[1] where we mention $p = \frac{a}{b}$

[2] p is unknown but expected to be around 1/3. Standard error will be approximated

$$SE = \sqrt(\frac{p(1-p)}{n}) \approx \sqrt{\frac{1/3(1-1/3)}{300}} = 0.027$$

Bibliography

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2022). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.30.