

Rohan Alexander

Telling Stories With Data

For Mum and Dad

Contents

List of Tables	xvii
List of Figures	xix
Preface	xxiii
About the author	xxxiii
1 Telling stories with data	3
1.1 On telling stories	3
1.2 Workflow components	4
1.3 Telling stories with data	7
1.4 How do our worlds become data?	12
1.5 What is data science and how should we use it to learn about the world	14
1.6 Exercises and tutorial	16
1.6.1 Exercises	16
1.6.2 Tutorial	17
2 Drinking from a fire hose	19
2.1 Hello, World!	20
2.2 Canadian elections	22
2.2.1 Plan	22
2.2.2 Simulate	22
2.2.3 Acquire	25
2.2.4 Explore	31
2.2.5 Communicate	32
2.3 Toronto homelessness	34
2.3.1 Plan	34
2.3.2 Simulate	35
2.3.3 Acquire	38
2.3.4 Explore	40
2.3.5 Communicate	41
2.4 Neonatal mortality	42
2.4.1 Plan	42
2.4.2 Simulate	44

2.4.3	Acquire	47
2.4.4	Explore	51
2.4.5	Communicate	51
2.5	Exercises and tutorial	53
2.5.1	Exercises	53
2.5.2	Tutorial	55
3	R essentials	57
3.1	Background	59
3.2	Broader impacts	60
3.3	R, R Studio, and R Studio Cloud	62
3.3.1	R	62
3.3.2	R Studio	63
3.3.3	R Studio Cloud	65
3.4	Getting started	65
3.5	The <code>dplyr</code> verbs	68
3.5.1	<code>select()</code>	69
3.5.2	<code>filter()</code>	71
3.5.3	<code>arrange()</code>	75
3.5.4	<code>mutate()</code>	78
3.5.5	<code>summarise()</code>	83
3.6	Base	86
3.6.1	<code>class()</code>	86
3.6.2	Simulating data	92
3.6.3	<code>function()</code> , <code>for()</code> , and <code>apply()</code>	94
3.7	Making graphs with <code>ggplot2</code>	96
3.8	Exploring the <code>tidyverse</code>	102
3.8.1	Importing data and <code>tibble()</code>	102
3.8.2	Dataset manipulation with joins and pivots	104
3.8.3	String manipulation and <code>stringr</code>	106
3.8.4	Factor variables and <code>forcats</code>	108
3.9	Exercises and tutorial	110
3.9.1	Exercises	110
3.9.2	Tutorial	112
4	Reproducible workflows	115
4.1	Introduction	116
4.2	R Markdown	120
4.2.1	Getting started	120
4.2.2	Essential commands	120
4.2.3	R chunks	121
4.2.4	Abstracts and PDF outputs	122
4.2.5	References	123
4.2.6	Cross-references	124
4.3	R projects and file structure	126

4.4	Git and GitHub	127
4.4.1	Git	128
4.4.2	Github	129
4.4.3	Using GitHub within R Studio with the Git pane	132
4.4.4	Using GitHub with <code>usethis</code>	133
4.5	Using R in practice	134
4.5.1	Dealing with errors	134
4.5.2	Reproducible examples	135
4.5.3	Mentality	136
4.5.4	Code comments and style	137
4.6	Exercises and tutorial	139
4.6.1	Exercises	139
4.6.2	Tutorial	142
4.6.3	Paper	142
I	Foundations	1
5	On writing	145
5.1	Introduction	146
5.2	Developing research questions	150
5.2.1	Data-first	150
5.2.2	Question-first	151
5.3	Writing	152
5.3.1	Title	153
5.3.2	Abstract	154
5.3.3	Introduction	156
5.3.4	Data	158
5.3.5	Model	161
5.3.6	Results	162
5.3.7	Discussion	162
5.3.8	Brevity, typos, and grammar	163
5.3.9	Rules	164
5.4	Exercises and tutorial	164
5.4.1	Exercises	164
5.4.2	Tutorial	166
6	Static communication	169
6.1	Introduction	171
6.2	Graphs	171
6.2.1	Bar charts	176
6.2.2	Scatterplots	186
6.2.3	Line plots	196
6.2.4	Histograms	199
6.2.5	Boxplots	205
6.3	Tables	208

6.3.1	Showing part of a dataset	209
6.3.2	Communicating summary statistics	213
6.3.3	Display regression results	215
6.4	Maps	217
6.4.1	Australian polling places	220
6.4.2	Toronto bike parking	224
6.4.3	Geocoding	227
6.5	Exercises and tutorial	228
6.5.1	Exercises	228
6.5.2	Tutorial	229
7	Interactive communication	231
7.1	Introduction	232
7.2	Making a website	232
7.2.1	Postcards	233
7.2.2	Distill	234
7.2.3	Blogdown	237
7.3	Interactive maps	239
7.3.1	Leaflet	241
7.3.2	Mapdeck	244
7.4	Shiny	247
7.5	Exercises and tutorial	250
7.5.1	Exercises	250
7.5.2	Tutorial	250
7.5.3	Paper	250
II	Communication	143
8	Gather data	253
8.1	APIs	256
8.2	Case study - arXiv	258
8.3	Case study - rtweet	259
8.4	Case study - spotiflyr	263
8.5	Scraping	269
8.5.1	Introduction	269
8.5.2	Getting started	270
8.6	Case study - Rohan's books	271
8.6.1	Introduction	271
8.6.2	Gather	271
8.6.3	Clean	272
8.6.4	Explore	277
8.7	Case study - Canadian Prime Ministers	278
8.7.1	Introduction	278
8.7.2	Gather	280
8.7.3	Clean	280

8.7.4	Explore	284
8.8	PDFs	285
8.8.1	Introduction	285
8.8.2	Getting started	286
8.9	Case-study: US Total Fertility Rate, by state and year (2000-2018)	290
8.9.1	Introduction	290
8.9.2	Begin with an end in mind	290
8.9.3	Start simple, then iterate.	291
8.9.4	Iterating	297
8.10	Semi-structured	303
8.10.1	JSON and XML	303
8.11	Optical Character Recognition	303
8.12	Text	305
8.12.1	Introduction	305
8.12.2	Getting text data	306
8.12.3	Preparing text datasets	307
8.13	Exercises and tutorial	312
8.13.1	Exercises	312
8.13.2	Tutorial	314
9	Hunt data	315
9.1	Experiments and randomised controlled trials	319
9.1.1	Introduction	319
9.1.2	Motivation and notation	320
9.1.3	Randomised sampling	322
9.1.4	ANOVA	326
9.1.5	Treatment and control	328
9.2	Case study - Fisher's tea party	332
9.3	Case study - Tuskegee Syphilis Study	337
9.4	Sampling and survey essentials	339
9.4.1	Introduction	339
9.4.2	Simple random sampling	342
9.4.3	Stratified and cluster sampling	342
9.4.4	Snowball sampling and confidant methods	342
9.5	Case study - The Oregon Health Insurance Experiment	342
9.6	Case study - Student Coaching: How Far Can Technology Go?	344
9.7	Case study - Civic Honesty Around The Globe	346
9.8	A/B testing	350
9.8.1	Introduction	350
9.8.2	Delivery	353
9.8.3	Instrumentation	353
9.8.4	Randomisation unit	354
9.8.5	Partnerships	355
9.8.6	Speed vs quality	356

9.8.7 Conflicting priorities	356
9.9 Case study - Upworthy	357
9.10 Implementing surveys	361
9.10.1 Google	361
9.10.2 Facebook	361
9.10.3 Survey Monkey	361
9.10.4 Mechanical Turk	361
9.10.5 Prolific	361
9.10.6 Qualtrics	361
9.10.7 Other	361
9.11 Sensor data	361
9.12 FOI	361
9.13 Next steps	361
9.14 Exercises and tutorial	362
9.14.1 Exercises	362
9.14.2 Tutorial	364
10 Farm data	367
10.1 Overview	367
10.2 Censuses	369
10.2.1 Canada	370
10.2.2 USA	371
10.2.3 Australia	372
10.3 Other government data	372
10.3.1 Unemployment	372
10.3.2 Weather	372
10.3.3 That Canadian survey that we used in STA304 from the library	372
10.4 Open Government Data	372
10.4.1 Canada	372
10.5 Electoral Studies	372
10.5.1 Canadian Electoral Study	372
10.5.2 Australian Electoral Study	372
10.5.3 CCES	372
10.6 Other	372
10.7 Exercises and tutorial	373
10.7.1 Exercises	373
10.7.2 Tutorial	373
III Measure and acquire	251
11 Cleaning and preparing data	377
11.1 Introduction	378
11.2 Workflow	380
11.2.1 Save a copy of the raw data	380

11.2.2 Begin with an end in mind	380
11.2.3 Start small	382
11.2.4 Write tests and documentation.	384
11.2.5 Iterate, generalize and update	387
11.3 Case study - Kenya census	387
11.3.1 Set-up	388
11.3.2 Extract	389
11.3.3 Clean	391
11.3.4 Check	396
11.3.5 Tidy-up	399
11.3.6 Make Monica's dataset	399
11.4 Checks and tests	400
11.4.1 Plots	401
11.4.2 Counts	402
11.4.3 Go/no-go	402
11.4.4 Class	404
11.5 Naming things	406
11.6 Exercises and tutorial	409
11.6.1 Exercises	409
11.6.2 Tutorial	409
12 Storing and retrieving data	411
12.1 Introduction	411
12.2 Plan	412
12.3 R Packages for data	413
12.4 Documentation	413
12.5 Exercises and tutorial	413
12.5.1 Exercises	413
12.5.2 Tutorial	414
13 Disseminating and protecting data	415
13.1 Introduction	415
13.2 What is personally identifying information	415
13.3 Hashing and salting	416
13.4 GDPR and HIPAA	416
13.5 Making fake data to distribute when you can't share the real stuff	416
13.6 Differential privacy	416
13.7 Exercises and tutorial	416
13.7.1 Exercises	416
13.7.2 Tutorial	416
IV Preparation	375
14 Exploratory data analysis	419
14.1 Introduction	422

14.2 Case study - TTC subway delays	423
14.2.1 Introduction	424
14.2.2 Gather the data	424
14.2.3 Sanity Checks	426
14.2.4 Missing values	427
14.2.5 Duplicate rows	429
14.2.6 Visualizing distributions	430
14.2.7 Groups and small counts	433
14.2.8 Visualizing time series	437
14.2.9 Visualizing relationships	439
14.2.10 Principal components analysis	441
14.3 Case study - Opinions about a casino in Toronto	445
14.3.1 Data preparation	445
14.3.2 Some visual exploration (and more cleanup, of course) .	449
14.3.3 Getting the data into a more model-suitable format .	451
14.3.4 Logistic Regression	455
14.4 Case study - Airbnb listing in Toronto	458
14.4.1 Essentials	458
14.4.2 Set up	458
14.4.3 Get data	458
14.4.4 Clean data	459
14.4.5 Explore data	478
14.4.6 Model data	481
14.5 Exercises and tutorial	488
14.5.1 Exercises	488
14.5.2 Tutorial	488
15 It's Just A Linear Model	489
15.1 Overview	492
15.2 Simple linear regression	494
15.2.1 Overview	494
15.2.2 Implementation in base R	499
15.2.3 Tidy up with broom	501
15.2.4 Testing hypothesis	505
15.2.5 On p-values	506
15.3 Multiple linear regression	507
15.3.1 Threats to validity and aspects to think about .	510
15.3.2 More credible outputs	511
15.3.3 Implementation in <code>tidymodels</code>	511
15.3.4 Implementation in <code>rstanarm</code>	513
15.4 Logistic regression	514
15.4.1 Overview	514
15.4.2 Implementation in base	514
15.4.3 Implementation in <code>tidymodels</code>	517
15.4.4 Implementation in <code>rstanarm</code>	518

<i>Contents</i>	xi
15.5 Poisson regression	519
15.5.1 Overview	519
15.5.2 Implementation in base	520
15.5.3 Implementation in <code>tidymodels</code>	521
15.6 Exercises and tutorial	522
15.6.1 Exercises	522
15.6.2 Tutorial	522
16 Causality from observational data	523
16.1 Introduction	531
16.2 DAGs and trying not to be tricked by the data	533
16.2.1 DAGs and confounding	533
16.2.2 Selection and measurement bias	538
16.2.3 Two common paradoxes	538
16.3 Difference in differences	541
16.3.1 Matching and difference-in-differences	541
16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974	553
16.4.1 Introduction	553
16.4.2 Background	554
16.4.3 Data	554
16.4.4 Model	557
16.4.5 Results	558
16.4.6 Other points	562
16.5 Case study - Funding of Clinical Trials and Reported Drug Efficacy	563
16.6 Regression discontinuity design	565
16.6.1 Introduction	565
16.6.2 Simulated example	566
16.6.3 Overlap	570
16.6.4 Examples	571
16.6.5 Implementation	576
16.6.6 Fuzzy RDD	578
16.6.7 Threats to validity	578
16.6.8 Weaknesses	581
16.7 Case study - Stiers, Hooghe, and Dassonneville, 2020	581
16.8 Case study - Caughey, and Sekhon., 2011	582
16.9 Instrumental variables	582
16.9.1 Introduction	582
16.9.2 History	584
16.9.3 Simulated example	584
16.9.4 Implementation	589
16.9.5 Assumptions	590
16.9.6 Conclusion	591
16.10 Case study - Effect of Police on Crime	592

16.10.1 Overview	592
16.10.2 Data	592
16.10.3 Model	592
16.10.4 Discussion	594
16.11 Exercises and tutorial	595
16.11.1 Exercises	595
16.11.2 Tutorial	595
17 Multilevel regression with post-stratification	597
17.1 Introduction	601
17.2 Simulation - Toddler bedtimes	604
17.2.1 Construct a population	604
17.2.2 Get a biased sample from it	607
17.2.3 Model the sample	609
17.2.4 Get a post-stratification dataset	609
17.2.5 Post-stratify our model estimates	611
17.3 Case study - Xbox paper	612
17.3.1 Overview	612
17.3.2 Model	613
17.3.3 Post-stratify	613
17.4 Simulation - Australian voting	615
17.4.1 Overview	615
17.4.2 Construct a survey	616
17.4.3 Model the survey	618
17.4.4 Post-stratify	621
17.4.5 Improving the model	622
17.5 Forecasting the 2020 US election	626
17.5.1 Survey data	626
17.5.2 Post-stratification data	630
17.5.3 Model	633
17.5.4 Post-stratify	635
17.6 Concluding remarks and next steps	637
17.7 Exercises and tutorial	638
17.7.1 Exercises	638
17.7.2 Tutorial	638
18 Text as data	639
18.1 Introduction	639
18.2 Lasso regression	640
18.3 Topic models	646
18.3.1 Overview	646
18.3.2 Document generation process	646
18.3.3 Analysis process	649
18.3.4 Warnings and extensions	651
18.4 Word embedding	651

<i>Contents</i>	xiii
18.5 Conclusion	651
18.6 Exercises and tutorial	652
18.6.1 Exercises	652
18.6.2 Tutorial	652
V Modelling	417
19 Using the cloud	655
19.1 Introduction	655
19.2 Google Colab	657
19.3 AWS	657
19.4 Google Compute Engine	658
19.5 Azure	658
19.6 Exercises and tutorial	659
19.6.1 Exercises	659
19.6.2 Tutorial	659
20 Deploying models	661
20.1 Introduction	662
20.2 Packages	662
20.3 Shiny	663
20.4 Plumber and model APIs	663
20.4.1 Hello Toronto	663
20.4.2 Local model	663
20.4.3 Cloud model	666
20.5 Exercises and tutorial	668
20.5.1 Exercises	668
20.5.2 Tutorial	668
21 Efficiency	669
21.1 Introduction	670
21.2 Data efficiency	670
21.2.1 SQL	670
21.2.2 Feather	670
21.3 Code efficiency	670
21.4 Code refactoring	670
21.4.1 Measure	670
21.5 Experimental efficiency	671
21.6 Other languages	671
21.6.1 Python	671
21.6.2 Julia	671
21.7 Exercises and tutorial	671
21.7.1 Exercises	671
21.7.2 Tutorial	671
22 Concluding remarks, open issues, next steps	673

22.1 Concluding remarks	673
22.2 Some issues	674
22.3 Next steps	675
Appendix	677
A Oh you think and shoulders and datasets	677
A.1 Oh, you think we have good data on that!	677
A.2 Shoulders of giants	678
A.3 Possible datasets	679
B Papers	681
B.1 Paper 1	681
B.1.1 Task	681
B.1.2 Guidance	681
B.1.3 Checks	682
B.1.4 FAQ	682
B.1.5 Rubric	683
B.1.6 Previous examples	685
B.2 Paper 2	686
B.2.1 Task	686
B.2.2 Guidance	686
B.2.3 Checks	689
B.2.4 FAQ	689
B.2.5 Rubric	689
B.3 ‘These numbers mean dial it up’	693
B.3.1 Task	693
B.3.2 Guidance	693
B.3.3 Checks	695
B.3.4 FAQ	696
B.3.5 Rubric	696
B.4 ‘Two Cathedrals’	700
B.4.1 Task	700
B.4.2 Guidance	700
B.4.3 Peer review submission	701
B.4.4 Conduct peer-review	701
B.4.5 Checks	702
B.4.6 FAQ	702
B.4.7 Rubric	702
B.4.8 Previous examples	706
B.5 ‘A Proportional Response’	706
B.5.1 Task	706
B.5.2 Recommended steps	706
B.5.3 Checks	708
B.5.4 FAQ	708

<i>Contents</i>	xv
B.6 ‘Mr Willis of Ohio’	708
B.6.1 Task	708
B.6.2 Recommended steps	708
B.6.3 Checks	710
B.6.4 FAQ	710
B.7 ‘Five Votes Down’	710
B.7.1 Task	710
B.7.2 Recommended steps	710
B.7.3 Checks	714
B.7.4 FAQ	714
B.8 ‘What’s next?’	714
B.8.1 Task	714
B.8.2 Recommended steps	714
B.8.3 Checks	718
B.8.4 FAQ	718
Bibliography	719



List of Tables

2.1	Homeless shelter usage in Toronto in 2021	41
4.1	Number of visits to the doctor in the past two weeks, based on the 1977–1978 Australian Health Survey	126
6.1	Mean and standard deviation for four 'datasaurus' datasets . .	173
6.2	Mean and standard deviation for Anscombe	176
6.3	First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US	211
6.4	First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US	211
6.5	First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US	212
6.6	First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US	213
6.7	First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US	214
6.8	Summary of categorical variables for the inflation and GDP dataset	214
6.9	Correlation between the variables for the inflation and GDP dataset	215
6.10	Descriptive statistics for the inflation and GDP dataset	215
6.11	Explaining GDP as a function of inflation	217
6.12	Two models of GDP as a function of inflation	218
15.1	511
15.2	517



List of Figures

1	Telling stories with data	xxiv
1.1	The workflow builds on various elements	8
1.2	This drawing is clearly a dog, even though it is just one line	11
2.1	Opening R Studio Cloud for the first time	21
2.2	Quick sketch of a dataset that could be useful for analysing Canadian elections	22
2.3	Quick sketch of a possible graph of the number of ridings won by each party	23
2.4	Number of seats won, by political party, at the 2019 Canadian Federal Election	33
2.5	Quick sketch of a dataset that could be useful for understanding shelter usage in Toronto	34
2.6	Quick sketch of a table of the average number of beds occupied each month	35
2.7	Quick sketch of a potentially useful NMR dataset	43
2.8	Quick sketch of a graph of NMR by country over time	43
2.9	Neonatal Mortality Rate (NMR), for Argentina, Australia, Canada, and Kenya, (1971-2020)	52
3.1	Probability, from XKCD	62
3.2	Opening R Studio for the first time	64
3.3	After running an R Script	67
4.1	Instructions given to Mechanical Turk workers for removing incomplete, non-Caucasian, nonadult, and nonhuman male faces	118
4.2	Workflow for telling stories with data	119
4.3	Number of doctor visits in the past two weeks, based on the 1977–1978 Australian Health Survey	122
4.4	Number of illnesses in the past two weeks, based on the 1977–1978 Australian Health Survey	125
4.5	An infamous response to the launch of Dropbox in 2007, trivializing the use-case for Dropbox, and while this user’s approach would probably work for them, it probably would not for most folks.	128
4.6	How to access the Terminal within R Studio	129

4.7 GitHub sign-up screen	130
4.8 Start process of creating a new repository	130
4.9 Finish creating a new repository	131
4.10 Get the details of your new repository	131
4.11 The Git pane in R Studio	133
5.1 Example of a well-captioned figure	160
5.2 Example of a well-captioned table	160
6.1 Graph of four ‘datasaurus’ datasets	174
6.2 Recreation of Anscombe’s Quartet	176
6.3 Distribution of ages in the 1997-2001 British Election Panel Study	178
6.4 Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study	178
6.5 Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study	179
6.6 Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study, illustrating different themes	181
6.7 Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study	181
6.8 Distribution of age by gender, and vote preference, in the 1997-2001 British Election Panel Study	182
6.9 Distribution of age by gender, and vote preference, in the 1997-2001 British Election Panel Study	183
6.10 Distribution of age by gender, and vote preference, in the 1997-2001 British Election Panel Study	184
6.11 Distribution of age and vote preference, in the 1997-2001 British Election Panel Study	186
6.12 Relationship between inflation and GDP for Australia, Ethiopia, India, and the US	190
6.13 Relationship between inflation and GDP for Australia, Ethiopia, India, and the US	190
6.14 Relationship between inflation and GDP for Australia, Ethiopia, India, and the US	192
6.15 Relationship between inflation and GDP for Australia, Ethiopia, India, and the US	193
6.16 Relationship between inflation and GDP for Australia, Ethiopia, India, and the US	194
6.17 Relationship between inflation and GDP for Australia, Ethiopia, India, and the US	196
6.18 GDP over time for Australia, Ethiopia, India, and the US	197
6.19 GDP over time for Australia, Ethiopia, India, and the US	198
6.20 GDP over time for Australia, Ethiopia, India, and the US	199
6.21 Phillips curve for the US (1960-2020)	200

6.22 Distribution of income	200
6.23 Distribution of GDP in the United States	201
6.24 Distribution of GDP in the United States	203
6.25 Distribution of GDP in the United States	204
6.26 Distribution of GDP in the United States	205
6.27 Data drawn from beta distributions with different parameters	207
6.28 Data drawn from beta distributions with different parameters	207
6.29 Data drawn from beta distributions with different parameters	208
6.30 Data drawn from beta distributions with different parameters	209
7.1 Example of a website made with ‘postcards’ using the ‘trestles’ theme	233
7.2 Example of Trestles website with updated details	234
7.3 Example settings for setting up ‘distill’	235
7.4 Example of default ‘distill’ website	235
7.5 Updating the yaml to change the homepage	236
7.6 Adding another page to the website	237
7.7 Example settings for setting up blogdown	238
7.8 Serving default blogdown site	238
7.9 Using the Apéro theme	239
7.10 Popular baby names	248
7.11 Example of Shiny app where the user controls the number of bins	249
8.1 Example of Google Maps, as at 25 January 2021.	257
8.2 rtweet authorisation page	261
8.3 rtweet authorisation page	264
8.4 Some of Rohan’s books	274
8.5 Source code for top of the page	275
8.6 Source code for list	275
8.7 Sketch of planned graph.	279
8.8 Using the Selector Gadget to identify the tag, as at 13 March 2020.	282
8.9 First sentence of Jane Eyre	287
8.10 First few paragraphs of Jane Eyre	288
8.11 Example Vital Statistics Report, from 2000	291
8.12 Desired output from the PDF	293
8.13 Scan of first page of Jane Eyre.	304
9.1 Afternoon Tea Party (1890–1891), by Mary Cassatt (American, 1844–1926), as downloaded from https://artvee.com/dl/afternoon-tea-party	332
9.2 ‘The triumph of wisdom over fortune’ by Otto van Veen (Flemish, 1556 – 1629), as downloaded from https://artvee.com/dl/the-triumph-of-wisdom-over-fortune	337

9.3 Example of the results of the intervention.	347
9.4 Example of the wallet.	348
9.5 Key finding as to wallet return rates.	349
9.6 Comparison of the average number of clicks when a headline contains a question mark or not.	360
11.1 Example of a dataset end plan	381
15.1 Oh my.	495
15.2 Don't leave out the main effects in an interactive model	510
15.3 Examining the type of gold some jewellery is made from.	515
16.1 Using a DAG to illustrate perceived relationships	534
16.2 Carrot as a confounder	535
16.3 Carrot as a mediator	536
16.4 Carrot as a collider	537
16.5 Angelucci and Cagé, 2019, summary statistics: national daily newspapers	556
16.6 Angelucci and Cagé, 2019, summary statistics: local daily newspapers	556
16.7 Summary statistics from Ooostrom	564
16.8 Results	565
16.9 The explosion of regression discontinuity designs in recent years.	566
16.10 Effect of minimum unit pricing for alcohol in Scotland.	570
16.11 It took four bounces to go in, so how different were the teams...?	571
16.12 Nobody expects the Spanish Inquisition.	572
16.13 George, 2019, descendant effects identified using close elections RD design (p. 41).	575
16.14 George, 2020, UFCo effect on the probability of being poor (p. 17).	577
16.15 Bunching around marathon times.	579
16.16 Summary statistics for Levitt 2002.	593
16.17 The relationship between firefighters, police and crime.	594
16.18 The impact of police on crime.	595
17.1 What does Bernie ask us to do?	605
17.2 Post-stratified estimates for each state based on the Xbox survey and MRP	614
17.3 Post-stratified estimates on a demographic basis based on the Xbox survey and MRP	614
17.4 Post-stratified estimates of electoral college outcomes based on the Xbox survey and MRP	615

Preface

This book will help you tell stories with data. It establishes a foundation on which you can build and share knowledge about an aspect of the world of interest to you based on data that you observe. Telling stories in small groups around a fire played a critical role in the development of humans and society (Wiessner, 2014). Today our stories, based on data, can influence millions.

In this book we will explore, prod, push, manipulate, knead, and ultimately, try to understand the implications of data. A variety of features drive the choices in this book.

The motto of the university from which I took my PhD is *naturam primum cognoscere rerum* or roughly ‘learn the first nature of things’. But the original quote continues *temporis aeterni quoniam*, or roughly ‘for eternal time’. We will do both of these things. I focus on tools, approaches, and workflows that enable you to establish lasting and reproducible knowledge.

When I talk of data in this book, it will typically be related to humans. Humans will be at the centre of most of our stories, and we will tell social, cultural, and economic stories. In particular, throughout this book I will draw attention to inequity both in social phenomena and in data. Most data analysis reflects the world as it is. Many of the least well-off face a double burden in this regard: not only are they disadvantaged, but the extent is more difficult to measure. Respecting those whose data are in our dataset is a primary concern, and so is thinking of those who are systematically not in our dataset.

While data are often specific to various contexts and disciplines, the approaches used to understand them tend to be similar. Data are also increasingly global with resources and opportunities available from a variety of sources. Hence, I draw on examples from many disciplines and geographies.

To become knowledge, our findings must be communicated to, understood, and trusted by other people. Scientific and economic progress can only be made by building on the work of others. And this is only possible if we can understand what they did. Similarly, if we are to create knowledge about the world, then we must enable others to understand precisely what we did, what we found, and how we went about our tasks. As such, in this book I will be particularly prescriptive about communication and reproducibility.

Improving the quality of quantitative work is an enormous challenge, yet it is

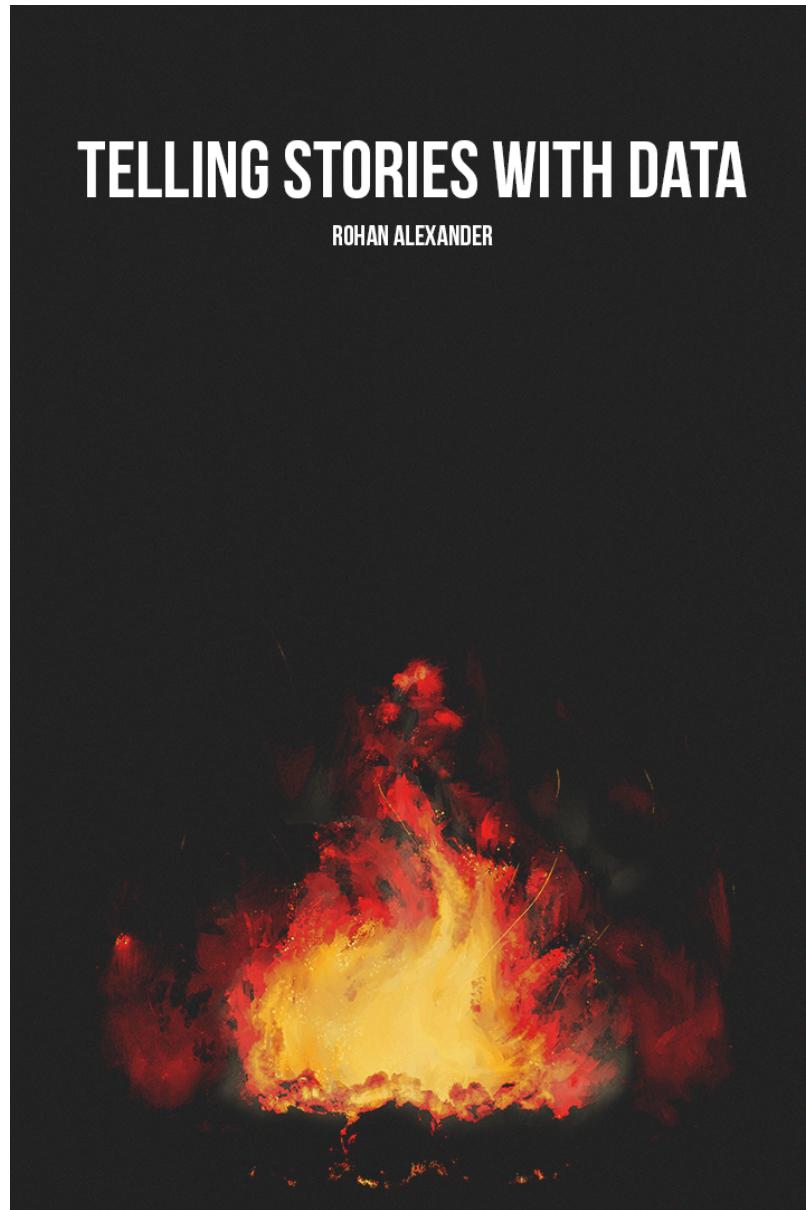


FIGURE 1: Telling stories with data

the challenge of our time. Data are all around us, but there is little enduring knowledge being created. This book hopes to contribute, in some small way, to changing that.

Audience and assumed background

The typical person reading this book has some familiarity with first-year statistics, for instance they have run a regression. But it is not targeted at a particular level, instead providing aspects relevant to almost any quantitative course. I have taught from this book at high school, undergraduate, and graduate levels. Everyone has unique needs, but hopefully some aspect of this book speaks to you.

This book especially complements *Statistical Rethinking* (McElreath, 2020), *R for Data Science* (Wickham and Grolemund, 2017), *An Introduction to Statistical Learning* (James et al., 2017), *Causal Inference: The Mixtape* (Cunningham, 2021), and *Building Software Together* (Wilson, 2021). For instance, after this book you may be interested in learning more about Bayesian statistics, data science, statistical learning, causal inference, and building software.

All of that said, some of the most successful students have been those with no quantitative or coding background. Enthusiasm and interest have taken folks far. If you have those, then don't worry about too much else.

Structure and content

This book is structured around six parts: I) Foundations, II) Communication, III) Acquisition, IV) Preparation, V) Modelling, and VI) Enrichment.

Part I – Foundations – begins with Chapter 1, which provides an overview of what I am trying to achieve with this book and why you should read it. Chapter 2 provides some worked examples. The intention of these is that you can experience the full workflow recommended in this book without worrying too much about the specifics of what is happening. That workflow is: plan, simulate, acquire, model, and communicate. It is normal to not follow everything in this chapter, but you should go through it, typing out and executing the code yourself. If you only have time to read one chapter of this book, then I recommend that one. Chapter 3 goes through some essential tasks in R, which is the statistical programming language used in this book. It is more of a reference chapter, and you may find yourself returning to it from time to time. And Chapter 4 introduces some key tools for reproducibility used in

the workflow that I advocate. These are things like using the command line, R Markdown, R Projects, Git and GitHub, and using R in practice.

Part II – Communication – considers three types of communication: written, static, and interactive. Chapter 5 details the features that quantitative writing should have and how to go about writing a crisp, technical, paper. Static communication in Chapter 6 introduces features like graphs, tables, and maps. Interactive communication in Chapter 7 covers aspects such as websites, web applications, and maps that can be manipulated.

Part III – Acquisition – focuses on three aspects: gathering data, hunting data, and farming data. Gathering data in Chapter 8 covers things like using Application Programming Interface (APIs), scraping data, getting data from PDFs, and Optical Character Recognition (OCR). The idea is that data are available, but not necessarily designed to be datasets, and that we must go and get them. Hunting data in Chapter 9 covers aspects where more is expected of us. For instance, we may need to conduct an experiment, run an A/B test, or do some surveys. Finally, farming data in Chapter 10 covers datasets that are explicitly provided for us to use as data, for instance censuses and other government statistics. These are typically clean, pre-packaged datasets.

Part IV – Preparation – covers how to respectfully transform raw data into something that can be explored and shared. Chapter 11 begins by detailing some principles to follow when approaching the task of cleaning and preparing data, and then goes through specific steps to take and checks to implement. Chapter 12 focuses on methods of storing and retrieving those datasets, including the use of R packages. Chapter 13 discusses considerations and steps to take when wanting to disseminate datasets as broadly as possible, while at the same time respecting those whose data they are based on.

Part V – Modelling – begins with exploratory data analysis in Chapter 14. This is the critical process of coming to understand the nature of a dataset, but not something that typically finds itself into the final product. In Chapter 15 the use of statistical models to explore data is introduced. Chapter 16 is the first of three applications of modelling. It focuses on attempts to make causal claims from observational data and covers approaches such as difference-in-differences, regression discontinuity, and instrumental variables. Chapter 17 is the second of the modelling applications chapters and focuses on multilevel regression with post-stratification where we use a statistical model to adjust a sample for known biases. Chapter 18 is the third and final modelling application and is focused on text-as-data.

Part VI – Enrichment – introduces various next steps that would improve aspects of the workflow and approaches introduced in previous chapters. Chapter 19 which goes through moving away from your own computer and toward using the cloud. Chapter 20 discusses deploying models through the use of packages, web applications, and APIs. Chapter 21 discusses various alterna-

tives to the storage of data including feather and SQL; and also covers some ways to improve the performance of your code. Finally, Chapter 22 offers some concluding remarks, details some open problems, and suggests some next steps.

Pedagogy and key features

You have to do the work. You should actively go through material and code yourself. As King (2000) says ‘[a]mateurs sit and wait for inspiration, the rest of us just get up and go to work’. Do not passively read this book. My role is best described by Hamming (1996, p. 2-3):

I am, as it were, only a coach. I cannot run the mile for you; at best I can discuss styles and criticize yours. You know you must run the mile if the athletics course is to be of benefit to you—hence you must think carefully about what you hear and read in this book if it is to be effective in changing you—which must obviously be the purpose...

This book is structured around a dense 12-week course. It provides enough material for advanced readers to be challenged, while establishing a core that all readers should master. Typically courses cover the material through to Chapter 15, and then pick another couple of chapters that are of particular interest.

From as early as Chapter 2 you will have a workflow—plan, simulate, acquire, model, and communicate—allowing you to tell a convincing story with data. In each subsequent chapter you will add aspects and depth to this workflow that will allow you to speak with increasing sophistication and credibility. As this workflow expands it addresses the skills that are typically sought in industry. For instance, features such as: communication, ethics, reproducibility, research question development, data collection, data cleaning, data protection and dissemination, exploratory data analysis, statistical modelling, and scaling.

One of the defining aspects of this book is that ethics and inequity concerns are integrated throughout, rather than being clustered in one, easily ignorable,

chapter. These aspects are critical, yet it can be difficult to immediately see their value, hence their tight integration.

This book is also designed to enable you to build a portfolio of work that you could show to a potential employer. If you want an industry job, then this is arguably the most important thing that you should be doing. Robinson and Nolis (2020, p. 55) describe how a portfolio is a collection of projects that show what you can do and is something that can help be successful in a job search.

In the novel *The Last Samurai* (DeWitt, 2000, p. 326), a character says:

[A] scholar should be able to look at any word in a passage and instantly think of another passage where it occurred; ... [so a] text was like a pack of icebergs each word a snowy peak with a huge frozen mass of cross-references beneath the surface.

In an analogous way, this book not only provides you with text and instruction that is self-contained, but also helps develop critical masses of knowledge on which expertise is built. No chapter positions itself as the last word, instead they are written in relation to other work.

Each chapter has the following features:

- A list of required materials that you should go through before you read that chapter. To be clear, you should first read that material and then return to this book. Each chapter also contains recommended materials for those who are particularly interested in the topic and want a starting place for further exploration.
- A summary of the key concepts and skills that are developed in that chapter. Technical chapters additionally contain a list of the main packages and functions that are used in the chapter. The combination of these features acts as a checklist for your learning, and you should return to them after completing the chapter.
- A series of short exercises that you should complete after going through the required materials, but before going through the chapter, to test your knowledge. After completing the chapter, you should go back through the exercises to make sure that you understand each aspect.
- One or two tutorial questions are included at the end of each chapter to further encourage you to actively engage with the material. You could consider forming small groups to discuss your answers to these questions.

Some chapters additionally feature:

- A section called ‘Oh, you think we have good data on that!’ which focuses on a particular setting, such as cause of death, in which it is often assumed that there is unimpeachable and unambiguous data but the reality tends to be quite far from that.
- A section called ‘Shoulders of giants’, which focuses on some of those who created the intellectual foundation on which we build.

Finally, a set of six papers is included in the appendix. If you write these, you will be conducting original research on a topic that is of interest to you. Although open-ended research may be new to you, the extent to which you are able to: develop your own questions, use quantitative methods to explore them, and communicates your findings, is the measure of the success of this book.

Software information and conventions

The software that I use in this book is R ([R Core Team, 2021](#)). This language was chosen because it is open source, widely used, general enough to cover the entire workflow, yet specific enough to have plenty of well-developed features. I do not assume that you have used R before, and so another reason for selecting R for this book is the community of R users. The community is especially welcoming of new-comers and there is a lot of complementary beginner-friendly material available. There is an R package, `DosSToolkit` ([Alexander et al., 2021](#)), that contains `learnr` modules ([Schloerke et al., 2021](#)). This may be useful if you are newer to R and are especially complementary to this book.

If you don’t have a programming language, then R is a great one to start with. If you have a preferred programming language already, then it wouldn’t hurt to pick up R as well. That said, if you have a good reason to prefer another open-source programming language (for instance you use Python daily at work) then you may wish to stick with that. However, all examples in this book are in R.

Please download R and R Studio onto your own computer. You can download R for free here: <http://cran.utstat.utoronto.ca/>, and you can download R Studio Desktop for free here: <https://rstudio.com/products/rstudio/download/#download>. Please also create an account on R Studio Cloud: <https://rstudio.cloud/>. This will allow you to run R in the cloud.

Packages are in typewriter text, for instance, `tidyverse`, while functions are also in typewriter text, but include brackets, for instance `dplyr::filter()`.

Acknowledgments

Many people generously gave code, data, examples, guidance, opportunities, thoughts, and time, that helped develop this book.

Thank you to David Grubbs and the team at CRC Press for taking a chance on me and providing invaluable support.

Thank you to Michael Chong and Sharla Gelfand for greatly helping to shape some of the approaches I advocate. However, both do much more than that and contribute in an enormous way to the spirit of generosity that characterises the R community.

Thank you to Kelly Lyons for her support, guidance, mentorship, and friendship. Every day she demonstrates what an academic should be, and more broadly, what we should all aspire to be as a person.

Thank you to Greg Wilson for providing a structure to think about teaching, for being the catalyst for this book, and for helpful comments on drafts. Every day he provides an example of how to contribute to the intellectual community.

Thank you to an anonymous reviewer and Isabella Ghement, who thoroughly went through an early draft of this book and provided detailed feedback that improved this book.

Thank you to Hareem Naveed for helpful feedback and encouragement. Her industry experience was an invaluable resource as I grappled with questions of coverage and focus.

Thank you to Leslie Root, who came up with the idea around ‘Oh, you think we have good data on that!’.

Thank you to my PhD supervisory panel John Tang, Martine Mariotti, Tim Hatton, and Zach Ward who gave me the freedom to explore the intellectual space that was of interest to me, the support to follow through on those interests, and the guidance to ensure that it all resulted in something tangible.

Thank you to Elle Côté for enabling this book to be written.

This book has greatly benefited from the notes and teaching materials of others that are freely available online, especially: Chris Bail, Scott Cunningham, Andrew Heiss, Lisa Lendway, Grant McDermott, Nathan Matias, David Mimno, and Ed Rubin. Thank you to these folks. The changed norm of academics making their materials freely available online is a great one and one that I hope the free online version of this book helps contribute to.

Thank you to those students who contributed substantially to the development of this book, including: A Mahfouz, Faria Khandaker, Keli Chiu, Paul

Hodgetts, and Thomas William Rosenthal. I discussed most aspects of this book with them, and while they made specific contributions, they also changed and sharpened the way that I thought about almost everything covered here. Paul additionally made the art for this book.

Thank you to those students who identified specific improvements, including: Aaron Miller, Amy Farrow, Cesar Villarreal Guzman, Flavia López, Hong Shi, Laura Cline, Lorena Almaraz De La Garza, Mounica Thanam, Reem Alasadi, Wijdan Tariq, and Yang Wu.

Finally, thank you to Monica Alexander. Without you I would not have written a book; I would not have even thought it possible. Thank you for your inestimable help with writing this book, providing the base on which it builds (remember in the library showing me many times how to get certain rows in R!), giving me the time that I needed to write, encouragement when it turned out that writing a book just meant endlessly re-writing that which was perfect the day before, reading everything in this book many times, providing perfect hydration in the form of coffee or cocktails as appropriate, and much more.

You can contact me at: rohan.alexander@utoronto.ca¹.

Rohan Alexander
Toronto, Canada

¹<mailto:rohan.alexander@utoronto.ca>



About the author

Rohan Alexander is an assistant professor at the University of Toronto in Information and Statistical Sciences. He is also the assistant director of CANSSI Ontario, a senior fellow at Massey College, a faculty affiliate at the Schwartz Reisman Institute for Technology and Society, and a co-lead of the Data Sciences Institute Thematic Program in Reproducibility. He holds a PhD in Economics from the Australian National University where he was supervised by John Tang (chair), Martine Mariotti, Tim Hatton, and Zach Ward.

He is interested in using statistical models to try to understand the world. And particularly how we get the data that go into those models; whose data are systematically missing; how we clean, prepare, and tidy data before they are modelled; the effects of all this on the implications of our models; and how we can reproducibly share the totality of this process. He tries to develop students that are skilled not only in using statistical methods across various disciplines, but also appreciate their limitations, and think deeply about the broader contexts of their work.

He enjoys teaching and aims to help students from a wide range of backgrounds learn how to use data to tell convincing stories. He teaches in both the Faculty of Information and the Department of Statistical Sciences at both undergraduate and graduate levels. He is a RStudio Certified Tidyverse Trainer.

He is married to Monica Alexander and they have two children. He probably spends too much money on books, and certainly too much time at libraries. If you have any book recommendations of your own, then he'd love to hear them.



Part I

Foundations



1

Telling stories with data

Required material

- Read *Counting the Countless*, (Keyes, 2019).
 - Watch *Data Science Ethics in 6 Minutes*, (Register, 2020).
-

1.1 On telling stories

Like many parents, when our children were born, one of the first things that my wife and I did regularly was read stories to them. In doing so we carried on a tradition that has occurred for millennia. Myths, fables, and fairy tales can be seen and heard all around us. Not only are they entertaining but they enable us to learn something about the world. While *The Very Hungry Caterpillar* (Carle, 1969) may seem quite far from the world of dealing with data, there are similarities. Both are trying to tell a story and teaching us something about the world.

When using data, we try to tell a convincing story. It may be as exciting as predicting elections, as banal as increasing internet advertising click rates, as serious as finding the cause of a disease, or as fun as forecasting basketball games. In any case the key elements are the same. The English author, E. M. Forster, described the aspects common to all novels as: story, people, plot, fantasy, prophecy, pattern, and rhythm (Forster, 1927). Similarly, when we tell stories with data, there are common concerns, regardless of the setting:

1. What is the dataset? Who generated the dataset and why?
2. What is the process that underpins the dataset? Given that process, what is missing from the dataset or been poorly measured? Could other datasets have been generated, and if so, how different could they have been to the one that we have?
3. What is the dataset trying to say, and how can we let it say this? What else could it say? How do we decide between these?
4. What are we hoping others will see from this dataset, and how can we convince them of this? How much work must we do to convince them? To what extent can we share how we came to believe this?

5. Who is affected by the processes and outcomes related to this dataset? To what extent are they represented in the dataset, and have they been part of conducting the analysis?

In the past, certain elements of telling stories with data were easier. For instance, experimental design has a long and robust tradition within agricultural and medical sciences, physics, and chemistry. Student's t-distribution was identified in the early 1900s by a chemist, William Sealy Gosset, who was working at Guinness, a beer manufacturer, and needed to assess the quality of the beer (Boland, 1984)! It would have been possible for him to randomly sample the beer and change one aspect at a time.

Many of the fundamentals of the statistical methods that we use today were developed in such settings. There, it was typically possible to establish control groups and randomize; and in these settings there were fewer ethical concerns. A story told with the resulting data is likely to be fairly convincing.

Unfortunately, little of this applies these days, given the diversity of settings to which statistical methods are applied. On the other hand, we have many advantages. For instance, we have well-developed statistical techniques, easier access to large datasets, and open-source statistical languages such as R (R Core Team, 2021). But the difficulty of conducting traditional experiments means that we must also turn to other aspects to tell a convincing story.

1.2 Workflow components

There are five core components to the workflow needed to tell stories with data:

1. **Plan** and sketch an endpoint.
2. **Simulate** some reasonable data and consider that.
3. **Acquire** and prepare the real data.
4. **Explore** and understand that dataset.
5. **Share** what we did and what we found.

We begin by **planning and sketching an endpoint** because this ensures that we think carefully about where we want to go. It forces us to deeply consider our situation, acts to keep us focused and efficient, and helps reduce scope creep. In *Alice's Adventures in Wonderland* (Carroll, 1865), Alice asks the Cheshire Cat which way she should go. The Cheshire Cat replies by asking where Alice would like to go. And when Alice replies that she does not mind, so long as she gets somewhere, the Cheshire Cat says then the direction does not matter because you will always get somewhere if you 'walk long enough'.

The issue, in our case, is that we typically cannot afford to walk aimlessly for long. While it may be that the endpoint needs to change, it is important that this is a deliberate, reasoned, decision. And that is only possible given an initial target. There is no need to spend too much time on this to get a lot of value from it. Often five minutes with paper and pen, is enough.

The next step is to **simulate data** because that forces us into the details. It helps with cleaning and preparing the dataset because it focuses us on the classes in the dataset and the distribution of the values that we expect. For instance, if we were interested in the effect of age-groups on political preferences, then we may expect that our age-group column would be a factor, with four possible values: ‘18-29’, ‘30-44’, ‘45-59’, ‘60+'. The process of simulation provides us with clear features that our real dataset should satisfy. We could use these features to define tests that would guide our data cleaning and preparation. For instance, we could check our real dataset for age-groups that are not one of those four values. When those tests pass, we could be confident that our age-group column only contains values that we expect.

Simulating data is also important when we turn to the statistical modelling stage. When we are at that stage, we are concerned with whether the model reflects what is in the dataset. The issue is that if we go straight to modelling the real dataset, then we do not know whether we have a problem with our model. We simulate data so that we precisely know the underlying data generation process. We then apply the model to that simulated dataset. When we get out what we put in, then we know that our model is performing appropriately, and can turn to the real dataset. Without that initial application to simulated data, it would be more difficult to have confidence in our model.

Simulation is often cheap—almost free given modern computing resources and statistical programming languages—and fast. It provides ‘an intimate feeling for the situation’ (Hamming, 1996, p. 239). The way to proceed is to start with a simulation that just contains the essentials, get that working, and to then complicate it.

Acquiring and preparing the data that we are interested in is an often-overlooked stage of the workflow. This is surprising because it can be one of the most difficult stages and requires many decisions to be made. This is increasingly the subject of research. For instance, it has been found that decisions made during this stage greatly affect statistical results (Huntington-Klein et al., 2021).

At this stage of the workflow, it is common to feel a little overwhelmed. Typically, the data we can acquire leave us a little scared. There may be too little of it, in which case we worry about how we are going to be able to make our statistical machinery work. Alternatively, we may have the other problem and be worried about how we can even begin to deal with such a large amount of data.

Perhaps all the dragons in our lives are princesses who are only waiting to see us act, just once, with beauty and courage. Perhaps everything that frightens us is, in its deepest essence, something helpless that wants our love.

Rilke (1929)

Developing comfort in this stage of the workflow unlocks the rest of it. The dataset that is needed to tell a convincing story is in there, but we need to iteratively remove everything that is not the data that we need, and to then shape that which is.

After we have a dataset, we then want to **explore and understand** certain relationships in that dataset. The use of statistical models to understand the implications of our data is not free of bias, nor are they ‘truth’; they do what we tell them to do. Within a workflow to tell stories with data, statistical models are tools and approaches that we use to explore our dataset, in the same way that we may use graphs and tables. They are not something that will provide us with a definitive result but will enable us to understand the dataset more clearly in a particular way.

By the time we get to this step in the workflow, to a large extent, the model will reflect the decisions that were made in earlier stages, especially acquisition and cleaning, as much as it reflects any type of underlying process. Sophisticated modelers know that their statistical models are like the bit of the iceberg above the surface; they build on, and are only possible due to, the majority that is underneath, in this case, the data. But when an expert at the whole workflow uses modelling, they recognize that the results that are obtained are additionally due to choices about whose data matters, decisions about how to measure and record the data, and other aspects that reflect the world as it is, well before that data is available to their specific workflow.

Finally, we must **share** what we did and what we found, at as high a fidelity as is possible. Talking about knowledge that only you have, does not make you knowledgeable, and that includes knowledge that only ‘past you’ has. When communicating, we need to be clear about what decisions we made, why we made them, our findings, and the weaknesses of our approach. We are aiming to uncover something important (otherwise, why bother) so we write down everything, in the first instance, although this written communication may be supplemented with other forms of communication later. There are so many decisions that we must make in this workflow that we want to be sure that we are open about the entire thing—start to finish. This means much more

than just the statistical modelling and creation of the graphs and tables, but everything. Without this, stories based on data do not have any credibility.

The world is not a rational meritocracy where everything is carefully and judiciously evaluated. Instead, we use shortcuts, hacks, and heuristics, based on our experience. Unclear communication will render even the best work moot, because it will not be thoroughly engaged with. While there is a minimum when it comes to communication, there is no upper limit to how impressive it can be. When it is the culmination of a thought-out workflow, at best, it obtains a certain *sprezzatura*, or studied carelessness. Achieving such mastery, is the work of years.

1.3 Telling stories with data

A compelling story based on data can likely be told in around ten-to-twenty pages. Much less than this, and it is likely too light on some of the details. And while it is easy to write much more, often some reflection enables succinctness or for multiple stories to be separated. The best stories are typically based on research and independent learning.

It is possible to tell convincing stories even when it is not possible to conduct traditional experiments. These approaches do not rely on ‘big data’—which is not a panacea (Meng, 2018)—but instead on better using the data that are available. A blend of theory and application, combined with practical skills, a sophisticated workflow, and an appreciation for what one does not know, is often enough to create lasting knowledge.

The best stories based on data tend to be multi-disciplinary. They take from whatever field they need to, but almost always draw on: statistics, data visualization, computer science, experimental design, economics, engineering, and information science (to name a few). As such, an end-to-end workflow requires a blend of skills from these areas. The best way to learn these skills is to use real-world data to conduct research projects where you:

- obtain and clean relevant datasets;
- develop research questions;
- use statistical techniques to explore those questions; and
- communicate in a meaningful way.

The key elements of telling convincing stories with data are:

1. Communication.
2. Ethics.
3. Reproducibility.

4. Questions.
5. Measurement.
6. Data collection.
7. Data cleaning.
8. Exploratory data analysis.
9. Modelling.
10. Scaling.

These elements are the foundation on which the workflow are built (Figure 1.1).

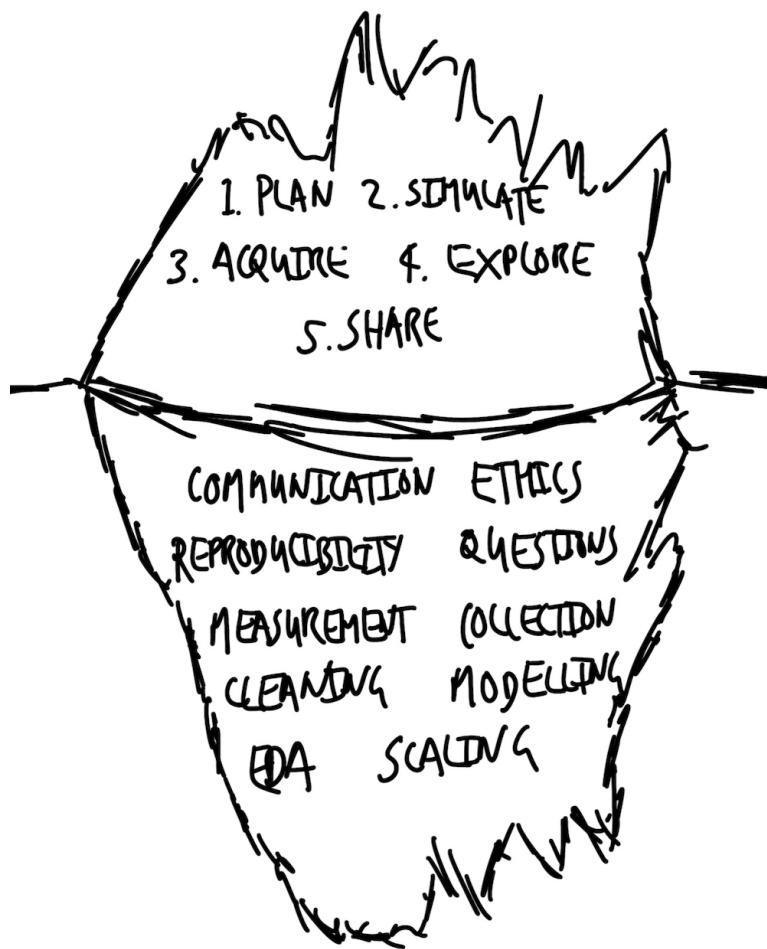


FIGURE 1.1: The workflow builds on various elements

This is a lot to master, but **communication** is the most important. Simple

analysis, communicated well, is more valuable than complicated analysis communicated poorly. This is because the latter cannot be understood or trusted by others. A lack of clear communication sometimes reflects a failure by the researcher to understand what is going on, or even what they are doing. And so, while the level of the analysis should match the dataset, instruments, task, and skillset, when a trade-off is required between clarity and complication, it can be sensible to err on the side of clarity.

Clear communication means writing in plain language, with the help of tables, graphs, and technical terms, in a way that brings the audience along with you. It means setting out what was done and why, as well as what was found. The minimum hurdle is doing this in a way that enables another person to independently do what you did and find what you found. One challenge is that as you immerse yourself in the data, it can be difficult to remember what it was like when you first came to it. But that is where most of your audience will be coming from. Learning to provide an appropriate level of nuance and detail is especially difficult but is made easier by trying to write for the audience's benefit.

Active consideration of **ethics** is needed because the dataset likely concerns humans. This means considering things like: who is in the dataset, who is missing, and why? To what extent will our story perpetuate the past? And is this something that ought to happen? Even if the dataset does not concern humans, the story is likely being put together by humans, and we affect almost everything else. This means we have a moral responsibility to use data ethically, with concern for environmental impact, and inequity.

There are many definitions of ethics, but when it comes to telling stories with data, at a minimum it means considering the full context of the dataset (D'Ignazio and Klein, 2020). In jurisprudence, a textual approach to law means literally considering the words of the law as they are printed, while a purposive approach means laws are interpreted within a broader context. An ethical approach to telling stories with data means adopting the latter approach, and considering the social, cultural, historical, and political forces that shape our world, and hence our data (Crawford, 2021).

Reproducibility is required to create lasting knowledge about the world. It means that everything that was done—all of it, end-to-end—can be independently redone. Ideally, autonomous end-to-end reproducibility is possible; anyone can get the code, data, and environment, to verify everything that was done. Unfettered access to code is almost always possible. While that is the default for data also, it is not always reasonable. For instance, studies in psychology may have small, personally identifying, samples. One way forward is to openly share simulated data with similar properties, along with defining a process by which the real data could be accessed, given appropriate *bona fides*.

Curiosity provides internal motivation to explore a dataset, and associated process, to a proper extent. **Questions** tend to beget questions, and these usually improve and refine as the process of coming to understand a dataset carries on. In contrast to the stock Popperian approach to hypothesis testing often taught, questions are typically developed through a continuous and evolving process (Franklin, 2005). It can be difficult to find an initial question. Selecting an area of interest can help, as can sketching a broad claim with the intent of evolving it into a specific question, and finally, bringing together two different areas.

Developing a comfort and ease in the messiness of real-world data means getting to ask new questions each time the data update. And knowing a dataset in detail tends to surface unexpected groupings or values that you can then work with subject-area experts to understand. Becoming a bit of a ‘mongrel’ by developing a base of knowledge across a variety of areas is especially valuable, as is becoming comfortable with the possibility of initially asking dumb questions.

Measurement and **data collection** are about deciding how our world will become data. They are challenging. The world is so vibrant that it is difficult to reduce it to something that is possible to consistently measure and collect. Take, for instance, someone’s height. We can, probably, all agree that we should take our shoes off before we measure height. But our height changes over the course of the day. And measuring someone’s height with a tape measure will give different results to using a laser. If we are comparing heights between people or over time, it therefore becomes important to measure at the same time each day, using the same method. But that quickly becomes unfeasible.

Most of the questions we are interested in will use data that are more complicated than height. How do we measure how sad someone is? How do we measure pain? Who decides what we will measure and how we will measure it? There is a certain arrogance required to think that we can reduce the world to a value and then compare these. Ultimately, we must, but it is difficult to consistently define what is to be measured. This process is not value-free. The only way to reasonably come to terms with this brutal reduction is to deeply understand, and respect what we are measuring and collecting. What is the central essence, and what can be stripped away?

Pablo Picasso, the twentieth century Spanish painter, has a series of drawings where he depicts the outline of an animal using only one line (Figure 1.2). Despite their simplicity, we recognize which animal is being depicted—the drawing is sufficient to tell the animal is a dog, not a cat. Could this be used to determine whether the dog is sick? Probably not. We would likely want a more detailed drawing. The decision as to which features should be measured and collected, and which to ignore, turns on context and purpose.



FIGURE 1.2: This drawing is clearly a dog, even though it is just one line

Data cleaning and preparation is a critical part of using data. We need to massage the data available to us into a dataset that we can use. This requires making a lot of decisions. The data cleaning and preparation stage is critical, and worthy of as much attention and care as any other.

Consider a survey that collected information about gender using four options: ‘man’, ‘woman’, ‘prefer not to say’, ‘other’, where ‘other’ dissolved into an open textbox. When we come to that dataset, we are likely to find that most responses are either ‘man’ or ‘woman’. We need to decide what to do about ‘prefer not to say’. If we drop it from our dataset, then we are actively ignoring these respondents. If we do not drop it, then it makes our analysis more complicated. Similarly, we need to decide how to deal with the open text responses. Again, we could drop these responses, but this ignores the experiences of some of our respondents. Another option is to merge this with ‘prefer not to say’, but that shows a disregard for our respondents, because they specifically did not choose that option.

There is no easy, nor always-correct, choice in many data cleaning and preparation situations. It depends on context and purpose. Data cleaning and preparation involves making many choices like this, and so it is vital to record every step so that others can understand what was done and why. Data never speak for themselves; they are the dummies of the ventriloquists that cleaned and prepared them.

The process of coming to understand the look and feel of a dataset is termed **exploratory data analysis** (EDA). This is an open-ended process. We need to understand the shape of our dataset before we can formally model it. The process of EDA is an iterative one that involves producing summary statistics, graphs, tables, and sometimes even some modelling. It is a process that never formally finishes and requires a variety of skills.

It is difficult to delineate where EDA ends and formal statistical modelling begins, especially when considering how beliefs and understanding develop

([Hullman and Gelman, 2021](#)). But at its core, ‘EDA starts from the data’, and involves immersing ourselves in it ([Cook et al., 2021](#)). EDA is not something that is typically explicitly part of our final story. But it has a central role in how we come to understand the story we are telling. And so, it is critical that all the steps taken during EDA are recorded and shared.

Statistical modelling has a long and robust history. Our knowledge of statistics has been built over hundreds of years. Statistics is not a series of dry theorems and proofs but is instead a way of exploring the world. It is analogous to ‘a knowledge of foreign languages or of algebra: it may prove of use at any time under any circumstances’ ([Bowley, 1901](#), p. 4). A statistical model is not a recipe to be blindly followed in an if-this-then-that way but is instead a way of understanding data ([James et al., 2017](#)). Modelling is usually required to infer statistical patterns from data. More formally, ‘statistical inference, or “learning” as it is called in computer science, is the process of using data to infer the distribution that generated the data’ ([Wasserman, 2005](#), p. 87).

Statistical significance is not the same as scientific significance, and we are realizing the cost of what has been the dominant paradigm. It is rarely appropriate to put our data through some arbitrary pass/fail statistical test. Instead, the proper use for statistical modelling is as a kind of echolocation. We listen to what comes back to us from the model, to help learn about the shape of the world, while recognizing that it is only one representation of the world.

The use of statistical programming languages, such as R, enables us to rapidly **scale** our work. This refers to both inputs and outputs. It is basically just as easy to consider 10 observations as 1,000, or even 1,000,000. This enables us to more quickly see the extent to which our stories apply. It is also the case that our outputs can be consumed as easily by one person as by 10, or 100. Using an Application Programming Interface (API) it is even possible for our stories to be considered many thousands of times each second.

1.4 How do our worlds become data?

There is the famous story by Eddington about some people who went fishing in the sea with a net. Upon examining the size of the fish they had caught, they decided there was a minimum size to the fish in the sea! Their conclusion arose from the tool used and not from reality.

Hamming (1996, p. 177)

To a certain extent we are wasting our time. We have a perfect model of the world—it is the world! But it is too complicated. If we knew perfectly how everything was affected by the uncountable factors that influence it, then we could perfectly forecast a coin toss, a dice roll, and every other seemingly random process each time. But we cannot. Instead, we must simplify things to that which is plausibly measurable, and it is that which we define as data. Our data are a simplification of the messy, complex, world from which they were generated.

There are different approximations of ‘plausibly measurable’. Hence, datasets are always the result of choices. We must decide whether they are nonetheless reasonable for the task at hand. We use statistical models to help us think deeply about, explore, and hopefully come to better understand, our data.

Much of statistics is focused on considering, thoroughly, the data that we have. And that was appropriate for when our data were predominately agricultural, astronomical, or from the physical sciences. But with the rise of data science, mostly because of the value of its application to datasets generated by humans, we must also actively consider what is not in our dataset. Who is systematically missing from our dataset? Whose data does not fit nicely into the approach that we are using and are hence being inappropriately simplified? If the process of the world becoming data necessarily involves abstraction and simplification, then we need to be clear about the points at which we can reasonably simplify, and those which would be inappropriate, recognizing that this will be application specific.

The process of our world becoming data necessarily involves measurement. Paradoxically, often those that do the measurement and are deeply immersed in the details have less trust in the data than those that are removed from it. Even seemingly clear tasks, such as measuring distance, defining boundaries, and counting populations, are surprisingly difficult in practice. Turning our world into data requires many decisions and imposes much error. Among many other considerations, we need to decide what will be measured, how accurately we will do this, and who will be doing the measurement.

Oh, you think we have good data on that! An important example of how something seemingly simple quickly becomes difficult is maternal mortality. That refers to the number of women who die while pregnant, or soon after a termination, from a cause related to the pregnancy or its management (WHO, 2019). It is

difficult but critical to turn the tragedy of such a death into cause-specific data because that helps mitigate future deaths. Some countries have well-developed civil registration and vital statistics (CRVS). These collect data about every death. But many countries do not have a CRVS and so not every death is recorded. Even if a death is recorded, defining a cause of death may be difficult, especially when there is a lack of qualified medical personal or equipment. Maternal mortality is especially difficult because there are typically many causes. Some CRVS have a checkbox on the form to specify whether the death should be counted as maternal mortality. But even some developed countries have only recently adopted this. For instance, it was only introduced in the US in 2003, and even in 2015 Alabama, California, and West Virginia had not adopted the standard question (MacDorman and Declercq, 2018).

We typically use various instruments to turn the world into data. In astronomy, the development of better telescopes, and eventually satellites and probes, enabled new understanding of other worlds. Similarly, we have new instruments for turning our own world into data being developed each day. Where once a census was a generational-defining event, now we have regular surveys, transactions data available by the second, and almost all interactions on the internet become data of some kind. The development of such instruments has enabled exciting new stories.

Our world imperfectly becomes data. If we are to nonetheless use data to learn about the world, then we need to actively seek to understand the ways they are imperfect and the implications of those imperfections.

1.5 What is data science and how should we use it to learn about the world

There is no agreed definition of data science, but a lot of people have tried. For instance, Wickham and Grolemund (2017) say it is ‘...an exciting discipline that allows you to turn raw data into understanding, insight, and knowledge.’ Similarly, Leek and Peng (2020) say it is ‘...the process of formulating a quantitative question that can be answered with data, collecting and cleaning the data, analyzing the data, and communicating the answer to the question to a relevant audience.’ Baumer et al. (2021) say it is ‘...the science of extracting meaningful information from data.’ Craiu (2019) argues that the lack of cer-

tainty as to what data science is might not matter because ‘...who can really say what makes someone a poet or a scientist?’ He goes on to broadly say that a data scientist is ‘...someone with a data driven research agenda, who adheres to or aspires to using a principled implementation of statistical methods and uses efficient computation skills.’

In any case, alongside specific, technical, definitions, there is value in having a simple definition, even if we lose a bit of specificity. Probability is often informally defined as ‘counting things’ (McElreath, 2020, p. 10). In a similar informal sense, data science can be defined as something like: humans measuring stuff, typically related to other humans, and using sophisticated averaging to explain and predict.

That may sound a touch cute, but Francis Edgeworth, the nineteenth century statistician and economist, considered statistics to be the science ‘of those Means which are presented by social phenomena’, so it is in good company (Edgeworth, 1885). In any case, one feature of this definition is that it does not treat data as *terra nullius*, or nobody’s land. Statisticians tend to see data as the result of some process that we can never know, but that we try to use data to come to understand. Many statisticians care deeply about data and measurement, but there are many cases in statistics where data kind of just appear; they belong to no one. But that is never actually the case.

Data is generated, and then must be gathered, cleaned, and prepared, and these decisions matter. Every dataset is *sui generis*, or a class by itself, and so when you come to know one dataset well, you just know one dataset, not all datasets.

Much of data science focuses on the ‘science’, but it is important to also focus on ‘data’. And that is another feature of that cutesy definition of data science. A lot of data scientists are generalists, who are interested in a broad range of problems. Often, the thing that unites these is the need to gather, clean, and prepare messy data. And often it is the specifics of those data that requires the most time, that updates most often, and that are worthy of our most full attention.

Jordan (2019) describes being in a medical office and being given some probability, based on prenatal screening, that his child, then a fetus, had Down syndrome. By way of background, one can test to know for sure, but that test comes with the risk of the fetus not surviving, so this initial screening probability matters. Jordan (2019) found those probabilities were being determined based on a study done a decade earlier in the UK. The issue was that in the ensuing 10 years, imaging technology had improved so the test was not expecting such high-resolution images and there had been a subsequent (false) increase in Down syndrome diagnoses when the images improved. There was no problem with the science, it was the data.

Shoulders of giants Dr Michael Jordan is Pehong Chen Distinguished Professor at the University of California, Berkeley. After taking a PhD in Cognitive Science from University of California, San Diego, in 1985, he was appointed as an assistant professor at MIT, being promoted to full professor in 1997, and in 1998 he moved to Berkeley. One area of his research is statistical machine learning. One particularly important paper is Blei et al. (2003), which enables text to be grouped together to define topics.

It is not just the ‘science’ bit that is hard, it is the ‘data’ bit as well. For instance, researchers went back and examined one of the most popular text datasets in computer science, and they found that around 30 per cent of the data were inappropriately duplicated (Bandy and Vincent, 2021). There is an entire field—linguistics—that specializes in these types of datasets, and inappropriate use of data is one of the dangers of any one field being hegemonic. The strength of data science is that it brings together folks with a variety of backgrounds and training to the task of learning about some dataset. It is not constrained by what was done in the past. But this means that we must go out of our way to show respect for those who do not come from our own tradition, but who are nonetheless as similarly interested in a dataset as we are. Data science is multi-disciplinary and increasingly critical; hence it must reflect our world. There is a pressing need a diversity of backgrounds, of approaches, and of disciplines in data science.

Our world is messy, and so are our data. To successfully tell stories with data you need to become comfortable with the fact that the process will be difficult. Hannah Fry, the British mathematician, describes spending six months rewriting code before it solved her problem (Thornhill, 2021). You need to learn to stick with it. You also need to countenance failure, and you do this by developing resilience and having intrinsic motivation. The world of data is about considering possibilities and probabilities, and learning to make trade-offs between them. There is almost never anything that we know for certain, and there is no perfect analysis.

Ultimately, we are all just telling stories with data, but these stories are increasingly among the most important in the world.

1.6 Exercises and tutorial

1.6.1 Exercises

1. According to [Register \(2020\)](#) data decisions affect (pick one)?
 - a. Real people.
 - b. No one.
 - c. Those in the training set.
 - d. Those in the test set.
2. What is data science (in your own words)?
3. According to [Keyes \(2019\)](#) what is perhaps a more accurate definition of data science (pick one)?
 - a. The inhumane reduction of humanity down to what can be counted.
 - b. The quantitative analysis of large amounts of data for the purpose of decision-making.
 - c. Data science is an inter-disciplinary field that uses scientific methods, processes, algorithms, and systems to extract knowledge and insights from many structural and unstructured data.
4. Imagine that you have a job in which including ‘race’ and/or sexuality as explanatory variables improves the performance of your model. What types of issues would you consider when deciding whether to include these variable in production (in your own words)?
5. Re-order the following steps of the workflow to be correct:
 1. Simulate.
 2. Acquire.
 3. Share.
 4. Plan.
 5. Explore.
6. According to [Crawford \(2021\)](#), which of the following forces shape our world, and hence our data (select all that apply)?
 - a. Political.
 - b. Historical.
 - c. Cultural.
 - d. Social.
7. What is required to tell convincing stories (select all that apply)?
 - a. Sophisticated workflow.
 - b. Practical skills.
 - c. Big data.
 - d. Humility about one’s own knowledge.
 - e. Theory and application.
8. Why is ethics a key element of telling convincing stories (in your own words)?

1.6.2 Tutorial

The purpose of this tutorial is to clarify in your mind the difficulty of measurement, even of seemingly simple things, and hence the likelihood of measurement issues in more complicated areas.

1. Please obtain some seeds for a fast-growing plant such as radishes, mustard greens, or arugula. Plant the seeds and measure how much soil you used. Water them and measure the water you used. Each day take a note of any changes. More generally, measure and record as much as you can. Note your thoughts about the difficulty of measurement. Eventually your seeds will sprout, and you should measure how big they are. We will return to use the data that you put together.
2. While you are waiting for the seeds to sprout, and for one week only, please measure the length of your hair daily. Write a one-or-two-page paper about what you found and what you learned about the difficulty of measurement.

2

Drinking from a fire hose

Required material

- Read *Data science as an atomic habit*, ([Barrett, 2021a](#))
- Read *This is how AI bias really happens—and why it's so hard to fix*, ([Hao, 2019](#))
- Read *The mundanity of excellence: An ethnographic report on stratification and Olympic swimmers*, ([Chambliss, 1989](#))

Key concepts and skills

- The statistical programming language R enables us to tell interesting stories using data. It is a language like any other, and the path to mastery can be slow.
- The framework that we use to approach projects is: plan, simulate, gather, explore, and share.
- The way to learn R is to start with a small project and break down what is required to achieve it into tiny steps, look at other people's code, and draw on that to achieve each step. Complete that project and move onto the next project. Each project you will get a little better.
- The key is to start actively working regularly.

Key libraries

- `ggplot2` ([Wickham, 2016](#))
- `janitor` ([Firke, 2020](#))
- `opendatatoronto` ([Gelfand, 2020](#))
- `tidyverse` ([Wickham, 2017](#))

Key functions

- `<- 'assign'`
- `|> 'pipe'`
- `+'add'`
- `c()`
- `citation()`
- `class()`
- `dplyr::arrange()`
- `dplyr::filter()`

- `dplyr::mutate()`
 - `dplyr::recode()`
 - `dplyr::rename()`
 - `dplyr::select()`
 - `dplyr::summarize()`
 - `ggplot2::geom_bar()`
 - `ggplot2::geom_point()`
 - `ggplot2::ggplot()`
 - `head()`
 - `janitor::clean_names()`
 - `library()`
 - `names()`
 - `readr::read_csv()`
 - `readr::write_csv()`
 - `rep()`
 - `rpois()`
 - `runif()`
 - `sample()`
 - `set.seed()`
 - `stringr::str_remove()`
 - `sum()`
 - `tail()`
 - `tidyverse::separate()`
-

2.1 Hello, World!

The way to start, is to start. In this chapter we go through three complete examples of the workflow advocated in this book. This means we will: plan, simulate, acquire, explore, and share. If you are new to R, then some of the code may be a bit unfamiliar to you. If you are new to statistics, then some of the concepts may be unfamiliar. Do not worry. It will all soon become familiar. The only way to learn how to tell stories, is to start telling stories yourself. This means that you should try to get these examples working. Do the sketches yourself, type everything out yourself (using R Studio Cloud if you are new to R and do not have it installed on your own computer), and execute it all. It is important, and normal, to realize that it will be challenging at the start.

Whenever you're learning a new tool, for a long time, you're

going to suck... But the good news is that is typical; that's something that happens to everyone, and it's only temporary.

Hadley Wickham as quoted by [Barrett \(2021a\)](#).

You will be guided thoroughly here. Hopefully by experiencing the power of telling stories with data, you will feel empowered to stick with it.

To get started, go to R Studio Cloud – <https://rstudio.cloud/> – and create an account. As we are not doing anything too involved the free version will be fine for now. Once you have an account and log in, then it should look something like Figure 2.1.

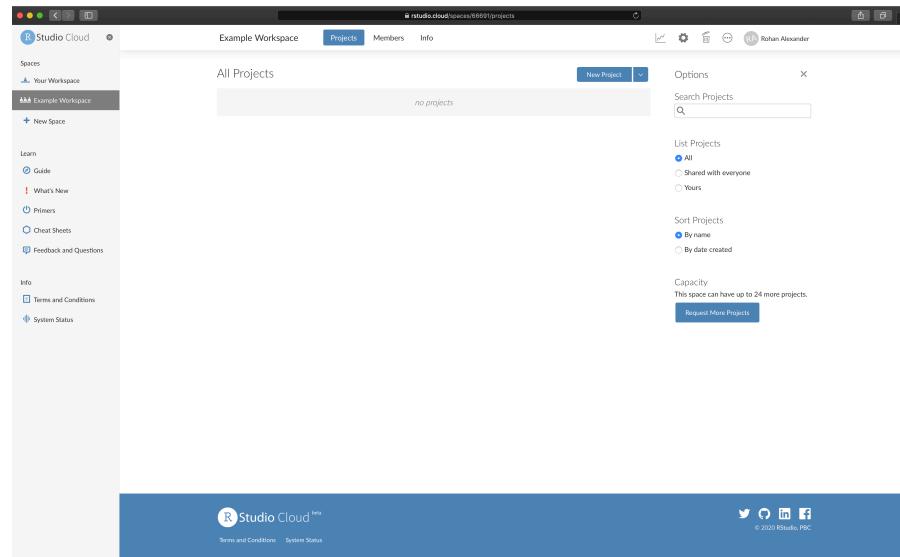


FIGURE 2.1: Opening R Studio Cloud for the first time

(You will be in ‘Your Workspace’, and you will not have an ‘Example Workspace’.) From here you should start a ‘New Project’. You can give the project a name by clicking on ‘Untitled Project’ and replacing it.

We will now go through three worked examples: Canadian elections, Toronto homelessness, and neonatal mortality. These examples build increasing complexity, but from the first one, we will be telling a story with data.

2.2 Canadian elections

Canada is a parliamentary democracy with 338 seats in the House of Commons, which is the lower house and that from which government is formed. There are two major parties – ‘Liberal’ and ‘Conservative’ – three minor parties – ‘Bloc Québécois’, ‘New Democratic’, and ‘Green’ – and many smaller parties and independents. In this example we will create a graph of the number of seats that each party won in the 2019 Federal Election.

2.2.1 Plan

For this example, we need to plan two aspects. The first is what the dataset that we need will look like, and the second is what the final graph will look like.

The basic requirement for the dataset is that it has the name of the seat (sometimes called a ‘riding’ in Canada) and the party of the person elected. So, a quick sketch of the dataset that we would need could look something like Figure 2.2.

RIDING	PARTY
---	LIB
---	CONS
---	LIB
:	:

FIGURE 2.2: Quick sketch of a dataset that could be useful for analysing Canadian elections

We also need to plan the graph that we are interested in. Given we want to display the number of seats that each party won, a quick sketch of what we might aim for is Figure 2.3.

2.2.2 Simulate

We now simulate some data, to bring some specificity to our sketches.



FIGURE 2.3: Quick sketch of a possible graph of the number of ridings won by each party

To get started, within R Studio Cloud, make a new R Markdown file ('File' -> 'New File' -> 'R Markdown'). When you do this, you will be asked to install some packages, which you should agree to. For this example, we will put everything into this one R Markdown document. You should save it as 'canadian_elections.Rmd' ('File' -> 'Save As...')

In the R Markdown document create a new R code chunk ('Code' -> 'Insert Chunk') and add preamble documentation that explains:

- the purpose of the document;
- the author and contact details;
- when the file was written or last updated; and
- pre-requisites that the file relies on.

In R, lines that start with '#' are comments. This means that they are not run as code by R, but are instead designed to be read by humans. Each line of this preamble should start with a '#'. Also make it clear that this is the preamble section by surrounding that with '#####'.

```
#### Preamble ####
# Purpose: Read in data from the 2019 Canadian Election and make a
# graph of the number of ridings each party won.
# Author: Rohan Alexander
# Email: rohan.alexander@utoronto.ca
# Date: 1 January 2022
# Prerequisites: Need to know where to get Canadian elections data.
```

After this we need to set-up the workspace. This involves installing and loading any packages that will be needed. A package only needs to be installed once for each computer, but needs to be loaded each time it is to be used. In this case we are going to use `tidyverse` (Wickham, 2017), `janitor` (Firke, 2020), and `tidyr` (Wickham, 2021b). They will need to be installed because this is the first time they are being used, and then each will need to be loaded.

An example of installing the packages follows (excessive comments have been added to be clear about what is going on; in general, this level of commenting is unnecessary). Run this code by clicking the small green arrow associated with the R code chunk.

```
#### Workspace set-up ####
install.packages("tidyverse") # Only need to do this once per computer
install.packages("janitor") # Only need to do this once per computer
install.packages("tidyr") # Only need to do this once per computer
```

Now that the packages are installed, they need to be loaded. As that installation step only needs to be done once per computer, that code can be commented out so that it is not accidentally run.

```
#### Workspace set-up ####
# install.packages("tidyverse") # Only need to do this once per computer
# install.packages("janitor") # Only need to do this once per computer
# install.packages("tidyr") # Only need to do this once per computer

library(tidyverse) # A collection of data-related packages
library(janitor) # Helps clean datasets
library(tidyr) # Helps make tidy datasets
```

All packages contain a help file that provides information about them and their functions. It can be accessed by appending a question mark before the package name and then running that code. For instance `?tidyverse`.

To simulate our data, we need to create a dataset with two columns: ‘Riding’ and ‘Party’, and some values for each. In the case of ‘Riding’ reasonable values

would be a name of one of the 338 Canadian ridings. In the case of ‘Party’ reasonable values would be one of the following six: ‘Liberal’, ‘Conservative’, ‘Bloc Québécois’, ‘New Democratic’, ‘Green’, ‘Other’. Again, this code can be run by clicking the small green arrow associated with the R code chunk.

```

simulated_data <-
  tibble(
    # Use 1 through to 338 to represent each riding
    'Riding' = 1:338,
    # Randomly choose one of six options, with replacement, 338 times
    'Party' = sample(
      x = c(
        'Liberal',
        'Conservative',
        'Bloc Québécois',
        'New Democratic',
        'Green',
        'Other'
      ),
      size = 338,
      replace = TRUE
    ))
  
```

```

simulated_data
#> # A tibble: 338 x 2
#>   Riding Party
#>   <int> <chr>
#> 1     1 Conservative
#> 2     2 New Democratic
#> 3     3 Liberal
#> 4     4 Bloc Québécois
#> 5     5 New Democratic
#> 6     6 Liberal
#> 7     7 Bloc Québécois
#> 8     8 Other
#> 9     9 Other
#> 10    10 Liberal
#> # ... with 328 more rows
  
```

2.2.3 Acquire

Now we want to get the actual data. The data we need is from Elections Canada, which is the non-partisan agency that organizes Canadian Federal elections. We can pass a website to `read_csv()` from the `readr` package ([Wick-](#)

ham et al., 2021). We do not need to explicitly load the `readr` package because it is part of the `tidyverse`. The ‘`<-`’ or ‘assignment operator’ is allocating the output of `read_csv()` to an object called ‘`raw_elections_data`’.

```
##### Read in the data #####
raw_elections_data <-
  read_csv(
    file =
      "https://www.elections.ca/res/rep/off/ovr2019app/51/data_donnees/table_tableau11.csv",
    show_col_types = FALSE
  )

# We have read the data from the Elections Canada website. We may like
# to save it in case something happens or they move it.
write_csv(
  x = raw_elections_data,
  file = "canadian_voting.csv"
)
```

We can take a quick look at the dataset using `head()` which will show the first six rows, and `tail()` which will show the last six rows.

```
head(raw_elections_data)
#> # A tibble: 6 x 13
#>   Province   `Electoral Distri~ `Electoral Distr~ Population
#>   <chr>       <chr>           <dbl>        <dbl>
#> 1 Newfoundl~ Avalon            10001        86494
#> 2 Newfoundl~ Bonavista--Burin--~ 10002        74116
#> 3 Newfoundl~ Coast of Bays--Ce~ 10003        77680
#> 4 Newfoundl~ Labrador          10004        27197
#> 5 Newfoundl~ Long Range Mounta~ 10005        86553
#> 6 Newfoundl~ St. John's East/S~ 10006        85697
#> # ... with 9 more variables: Electors/Électeurs <dbl>,
#> #   Polling Stations/Bureaux de scrutin <dbl>,
#> #   Valid Ballots/Bulletins valides <dbl>,
#> #   Percentage of Valid Ballots /Pourcentage des bulletins valides <dbl>,
#> #   Rejected Ballots/Bulletins rejetés <dbl>,
#> #   Percentage of Rejected Ballots /Pourcentage des bulletins rejetés <dbl>,
#> #   Total Ballots Cast/Total des bulletins déposés <dbl>, ...
tail(raw_elections_data)
#> # A tibble: 6 x 13
#>   Province   `Electoral Distri~ `Electoral Distr~ Population
#>   <chr>       <chr>           <dbl>        <dbl>
#> 1 British C~ Vancouver South/V~      59040       102927
```

```
#> 2 British C~ Victoria           59041   117133
#> 3 British C~ West Vancouver--S~ 59042   119113
#> 4 Yukon      Yukon            60001   35874
#> 5 Northwest~ Northwest Territo~ 61001   41786
#> 6 Nunavut    Nunavut          62001   35944
#> # ... with 9 more variables: Electors/Électeurs <dbl>,
#> #   Polling Stations/Bureaux de scrutin <dbl>,
#> #   Valid Ballots/Bulletins valides <dbl>,
#> #   Percentage of Valid Ballots /Pourcentage des bulletins valides <dbl>,
#> #   Rejected Ballots/Bulletins rejetés <dbl>,
#> #   Percentage of Rejected Ballots /Pourcentage des bulletins rejetés <dbl>,
#> #   Total Ballots Cast/Total des bulletins déposés <dbl>, ...
```

We need to clean the data so that we can use it. We are trying to make it similar to the dataset that we thought we wanted in the planning stage. While it is fine to move away from the plan, this needs to be a deliberate, reasoned, decision. After reading in the dataset that we saved, the first thing that we will do is adjust the names to make them easier to type. Removing the spaces helps to type column names. We will do this using `clean_names()` from `janitor` (Firke, 2020) which changes the names into ‘snake_case’.

```
##### Basic cleaning #####
raw_elections_data <-
  read_csv(file = "canadian_voting.csv",
           show_col_types = FALSE
  )

# Make the names easier to type
cleaned_elections_data <-
  clean_names(raw_elections_data)

# Have a look at the first six rows
head(cleaned_elections_data)
#> # A tibble: 6 x 13
#>   province electoral_distric~ electoral_distri~ population
#>   <chr>     <chr>                  <dbl>       <dbl>
#> 1 Newfoundl~ Avalon                10001     86494
#> 2 Newfoundl~ Bonavista--Burin--~ 10002     74116
#> 3 Newfoundl~ Coast of Bays--Ce~ 10003     77680
#> 4 Newfoundl~ Labrador              10004     27197
#> 5 Newfoundl~ Long Range Mounta~ 10005     86553
#> 6 Newfoundl~ St. John's East/S~ 10006     85697
```

```
#> # ... with 9 more variables: electors_electeurs <dbl>,
#> #   polling_stations_bureaux_de_scrutin <dbl>,
#> #   valid_ballots_bulletins_valides <dbl>,
#> #   percentage_of_valid_ballots_pourcentage_des_bulletins_valides <dbl>,
#> #   rejected_ballots_bulletins_rejetes <dbl>,
#> #   percentage_of_rejected_ballots_pourcentage_des_bulletins_rejetes <dbl>,
#> #   total_ballots_cast_total_des_bulletins_deposes <dbl>, ...
```

The names are faster to type because R Studio will auto-complete them. To do this, we begin typing the name of a column and then use ‘tab’ to auto-complete it.

There are many columns in the dataset, and we are primarily interested in two: ‘electoral_district_name_nom_de_circonscription’, and ‘elected_candidate_candidat_elu’. We can choose certain columns of interest using `select()` from `dplyr` (Wickham et al., 2020a) which we loaded as part of the `tidyverse`. The ‘pipe operator’, `|>`, pushes the output of one line to be the first input of the function on the next line.

```
cleaned_elections_data <-
  cleaned_elections_data |>
  # Select only certain columns
  select(electoral_district_name_nom_de_circonscription,
         elected_candidate_candidat_elu
         )

# Have a look at the first six rows
head(cleaned_elections_data)
#> # A tibble: 6 x 2
#>   electoral_district_name_no~ elected_candidate_candidat_elu
#>   <chr>                      <chr>
#> 1 Avalon                     McDonald, Kenneth Liberal/Lib~
#> 2 Bonavista--Burin--Trinity  Rogers, Churence Liberal/Libé~
#> 3 Coast of Bays--Central--No~ Simms, Scott Liberal/Libéral
#> 4 Labrador                    Jones, Yvonne Liberal/Libéral
#> 5 Long Range Mountains       Hutchings, Gudie Liberal/Libé~
#> 6 St. John's East/St. John's~ Harris, Jack NDP-New Democrat~
```

Some of the names of the columns are still quite long because they have both English and French in them. We can look at the names of the columns with `names()`. And we can change the names using `rename()` from `dplyr` (Wickham et al., 2020a).

```

names(cleaned_elections_data)
#> [1] "electoral_district_name_nom_de_circonscription"
#> [2] "elected_candidate_candidat_elu"

cleaned_elections_data <-
  cleaned_elections_data |>
  rename(
    riding = electoral_district_name_nom_de_circonscription,
    elected_candidate = elected_candidate_candidat_elu
  )

head(cleaned_elections_data)
#> # A tibble: 6 × 2
#>   riding                           elected_candidate
#>   <chr>                            <chr>
#> 1 Avalon                          McDonald, Kenneth Libera...
#> 2 Bonavista--Burin--Trinity       Rogers, Churence Liber...
#> 3 Coast of Bays--Central--Notre Dame Simms, Scott Liberal/L...
#> 4 Labrador                         Jones, Yvonne Liberal/...
#> 5 Long Range Mountains            Hutchings, Gudie Liberal...
#> 6 St. John's East/St. John's-Est Harris, Jack NDP-New D...

```

We will now look at this dataset, and the ‘elected_candidate’ column in particular.

```

head(cleaned_elections_data$elected_candidate)
#> [1] "McDonald, Kenneth Liberal/Libéral"
#> [2] "Rogers, Churence Liberal/Libéral"
#> [3] "Simms, Scott Liberal/Libéral"
#> [4] "Jones, Yvonne Liberal/Libéral"
#> [5] "Hutchings, Gudie Liberal/Libéral"
#> [6] "Harris, Jack NDP-New Democratic Party/NPD-Nouveau Parti démocratique"

```

We can see that we have the surname of the elected candidate, followed by a comma, followed by their first name, followed by a space, followed by the name of the party in both English and French, separated by a slash. We can break-up this column into its pieces using `separate()` from `tidyverse` (Wickham, 2021b).

```

cleaned_elections_data <-
  cleaned_elections_data |>
  # Separate the column into two based on the slash

```

```

separate(col = elected_candidate,
         into = c('other', 'party'),
         sep = '/') |>
# Remove the 'other' column
select(-other)

head(cleaned_elections_data)
#> # A tibble: 6 x 2
#>   riding                  party
#>   <chr>                   <chr>
#> 1 Avalon                  Libéral
#> 2 Bonavista--Burin--Trinity  Libéral
#> 3 Coast of Bays--Central--Notre Dame  Libéral
#> 4 Labrador                Libéral
#> 5 Long Range Mountains    Libéral
#> 6 St. John's East/St. John's-Est  NPD-Nouveau Parti démo~

```

Finally we want to change the party names from French to English to match what we simulated, using `recode()` from `dplyr` (Wickham et al., 2020a).

```

cleaned_elections_data <-
  cleaned_elections_data |>
  mutate(
    party =
      recode(
        party,
        'Conservateur' = 'Conservative',
        'Indépendant(e)' = 'Other',
        'Libéral' = 'Liberal',
        'NPD-Nouveau Parti démocratique' = 'New Democratic',
        'Parti Vert' = 'Green'
      )
  )

head(cleaned_elections_data)
#> # A tibble: 6 x 2
#>   riding                  party
#>   <chr>                   <chr>
#> 1 Avalon                  Liberal
#> 2 Bonavista--Burin--Trinity  Liberal
#> 3 Coast of Bays--Central--Notre Dame  Liberal
#> 4 Labrador                Liberal

```

```
#> 5 Long Range Mountains           Liberal
#> 6 St. John's East/St. John's-Est New Democratic
```

Our data now matches our plan (Figure 2.2) pretty well. For every electoral district we have the party of the person that won it.

Having now nicely cleaned the dataset, we should save it, so that we can start with that cleaned dataset in the next stage.

```
write_csv(
  x = cleaned_elections_data,
  file = "cleaned_elections_data.csv"
)
```

2.2.4 Explore

At this point we would like to explore the dataset that we created. One way to better understand a dataset is to make a graph. In particular, here we would like to build the graph that we planned in Figure 2.3.

First, we read in the dataset that we just created.

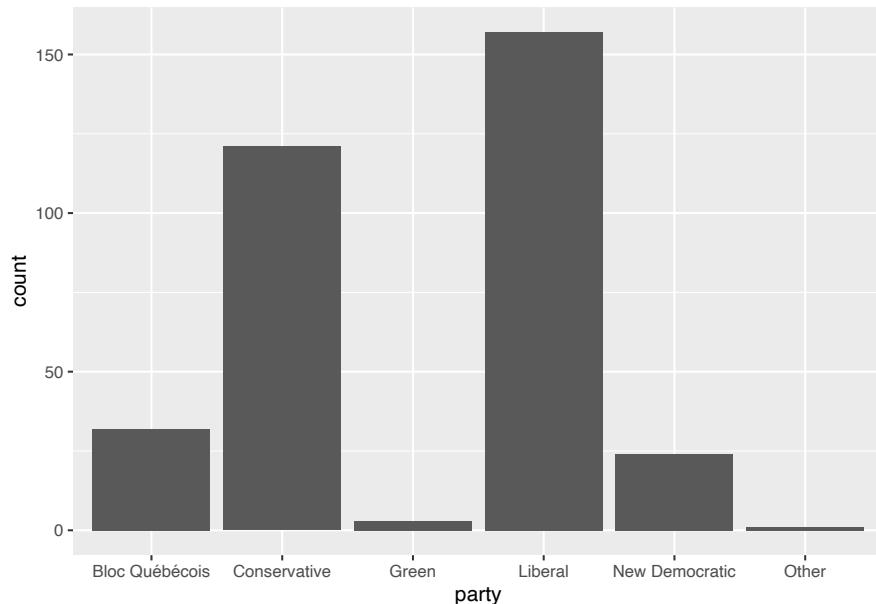
```
#### Read in the data ####
cleaned_elections_data <-
  read_csv(
    file = "cleaned_elections_data.csv",
    show_col_types = FALSE
  )
```

We can get a quick count of how many seats each party won using `count()` from `dplyr` (Wickham et al., 2020a).

```
cleaned_elections_data |>
  count(party)
#> # A tibble: 6 x 2
#>   party          n
#>   <chr>        <int>
#> 1 Bloc Québécois     32
#> 2 Conservative      121
#> 3 Green              3
#> 4 Liberal            157
#> 5 New Democratic     24
#> 6 Other               1
```

To build the graph that we are interested in, we will rely on the `ggplot2` package (Wickham, 2016). The key aspect of this package is that we build graphs by adding layers using ‘+’, which we call the ‘add operator’. In particular we will create a bar chart using `geom_bar()` from `ggplot2` (Wickham, 2016).

```
cleaned_elections_data |>
  ggplot(aes(x = party)) + # aes abbreviates aesthetics and enables us
  # to specify the x axis variable
  geom_bar()
```



This accomplishes what we set out to do. But we can make it look a bit nicer by modifying the default options (Figure 2.4).

```
cleaned_elections_data |>
  ggplot(aes(x = party)) +
  geom_bar() +
  theme_minimal() + # Make the theme neater
  coord_flip() + # Swap the x and y axis to make parties easier to read
  labs(x = "Party",
       y = "Number of seats") # Make the labels more meaningful
```

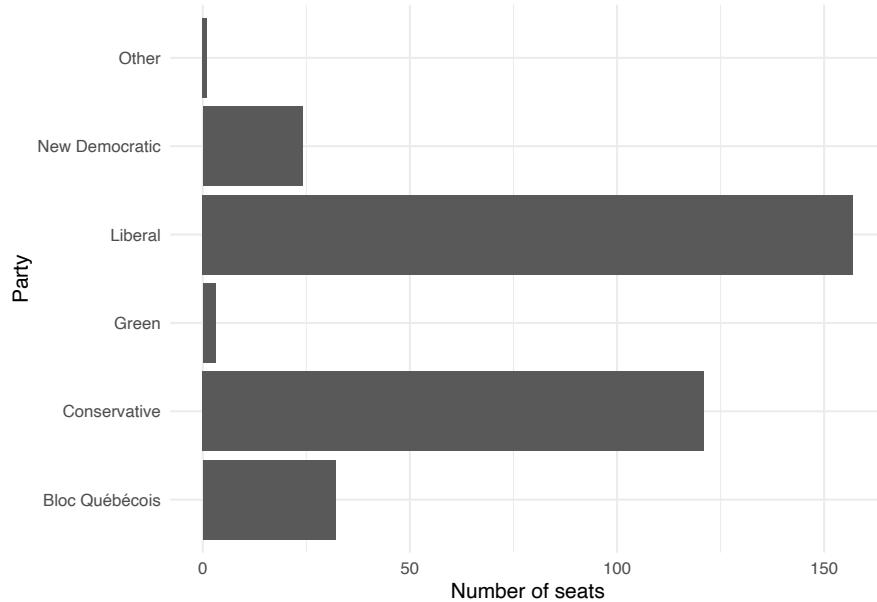


FIGURE 2.4: Number of seats won, by political party, at the 2019 Canadian Federal Election

2.2.5 Communicate

To this point we have downloaded some data, cleaned it, and made a graph. We would typically need to communicate what we have done at some length. In this case, we can write a few paragraphs about what we did, why we did it, and what we found. An example follows.

Canada is a parliamentary democracy with 338 seats in the House of Commons, which is the house that forms government. There are two major parties—‘Liberal’ and ‘Conservative’—three minor parties—‘Bloc Québécois’, ‘New Democratic’, and ‘Green’—and many smaller parties. The 2019 Federal Election occurred on 21 October, and more than 17 million votes were cast. We were interested in the number of seats that were won by each party.

We downloaded the results, on a seat-specific basis, from the Elections Canada website. We cleaned and tidied the dataset using the statistical programming language R ([R Core Team, 2021](#)) as well as the packages `tidyverse` ([Wickham et al., 2019a](#)) and

janitor (Firke, 2020). We then created a graph of the number of seats that each political party won (Figure 2.4).

We found that the Liberal Party won 157 seats, followed by the Conservative Party with 121 seats. The minor parties won the following number of seats: Bloc Québécois, 32 seats, New Democratic Party, 24 seats, and the Green Party, 3 seats. Finally, one independent candidate won a seat.

The distribution of seats is skewed toward the two major parties which could reflect relatively stable preferences on the part of Canadian voters, or possibly inertia due to the benefits of already being a major party such a national network and funding, or some other reason. A better understanding the reasons for this distribution are of interest in future work. While the dataset consists of everyone who voted, it worth noting that in Canada some are systematically excluded from voting; and it is much difficult for some to vote than others.

2.3 Toronto homelessness

Toronto has a large homeless population (City of Toronto, 2021). Freezing winters mean it is important there are enough places in shelters. In this example we will make a table of shelter usage in the second half of 2021 that compares average use in each month. Our expectation is that there is greater usage in the colder months, for instance, December, compared with warmer months, for instance, July.

2.3.1 Plan

The dataset that we are interested in would need to have date, the shelter, and the number of beds that were occupied that night. A quick sketch of a dataset that would work is Figure 2.5.

We are interested in creating a table that has the monthly average number of beds occupied each night. The table would probably look something like Figure 2.6.

2.3.2 Simulate

The next step is to simulate some data that could resemble our dataset.

date	shelter	occupancy
2021-07-01	—	27
2021-07-01	—	35
⋮	⋮	⋮

FIGURE 2.5: Quick sketch of a dataset that could be useful for understanding shelter usage in Toronto

MONTH	AVERAGE OCCUPANCY
JULY	500
AUGUST	475
⋮	⋮
DECEMBER	1000

FIGURE 2.6: Quick sketch of a table of the average number of beds occupied each month

Within R Studio Cloud make a new R Markdown file, save it, and make a new R code chunk and add preamble documentation. Then install and/or load the libraries that are needed. We will again use `tidyverse` (Wickham, 2017), `janitor` (Firke, 2020), and `tidyr` (Wickham, 2021b). As those were installed earlier, they do not need to be installed again. In this example we will also use `lubridate` (Grolemund and Wickham, 2011), which is part of the `tidyverse` and so it does not need to be installed independently. We will also use `opendatatoronto` (Gelfand, 2020), and `knitr` (Xie, 2021) and these will need to be installed.

```
#### Preamble ####
# Purpose: Get data about 2021 houseless shelter usage and make a table
# Author: Rohan Alexander
# Email: rohan.alexander@utoronto.ca
# Date: 1 January 2022
# Prerequisites: -

#### Workspace set-up ####
install.packages("opendatatoronto")
```

```
install.packages("lubridate")
install.packages("knitr")

library(knitr)
library(janitor)
library(lubridate)
library(opendatatoronto)
library(tidyverse)
library(tidyr)
```

To add a bit more detail to the earlier example, libraries contain code that other people have written. There are a few common ones that you will see regularly, especially the `tidyverse`. To use a package, we must first install it and then we need to load it. A package only needs to be installed once per computer but must be loaded every time. So, the packages that we installed earlier do not need to be reinstalled here.

Given that folks freely gave up their time to make R and the packages that we use, it is important to cite them. To get the information that is needed, we can use `citation()`. When run without any arguments, that provides the citation information for R itself, and when run with an argument that is the name of a package, it provides the citation information for that package.

```
citation() # Get the citation information for R
#>
#> To cite R in publications use:
#>
#>   R Core Team (2021). R: A language and environment
#>   for statistical computing. R Foundation for
#>   Statistical Computing, Vienna, Austria. URL
#>   https://www.R-project.org/.
#>
#> A BibTeX entry for LaTeX users is
#>
#> @Manual{,
#>   title = {R: A Language and Environment for Statistical Computing},
#>   author = {{R Core Team}},
#>   organization = {R Foundation for Statistical Computing},
#>   address = {Vienna, Austria},
#>   year = {2021},
#>   url = {https://www.R-project.org/},
#> }
#>
#> We have invested a lot of time and effort in creating
```

```
#> R, please cite it when using it for data analysis.
#> See also 'citation("pkgname")' for citing R packages.
citation('tidyverse') # Get the citation information for a particular package
#>
#> Wickham et al., (2019). Welcome to the tidyverse.
#> Journal of Open Source Software, 4(43), 1686,
#> https://doi.org/10.21105/joss.01686
#>
#> A BibTeX entry for LaTeX users is
#>
#> @Article{,
#>   title = {Welcome to the {tidyverse}},
#>   author = {Hadley Wickham and Mara Averick and Jennifer Bryan and Winston Chang and Lucy D'A
#>   year = {2019},
#>   journal = {Journal of Open Source Software},
#>   volume = {4},
#>   number = {43},
#>   pages = {1686},
#>   doi = {10.21105/joss.01686},
#> }
#>
```

Turning to the simulation, we need three columns: ‘date’, ‘shelter’, and ‘occupancy’. This example will build on the earlier one by adding a seed using `set.seed()`. A seed enables us to always generate the same random data. Any integer can be used as the seed. In this case the seed will be 853. If you use that as your seed, then you should get the same random numbers as in this example. If you use a different seed, then you should expect different random numbers. Finally, we use `rep()` to repeat something a certain number of times. For instance, we repeat ‘Shelter 1’, 184 times which accounts for half a year.

```
#### Simulate ####
set.seed(853)

simulated_occupancy_data <-
tibble(
  date = rep(x = as.Date("2021-07-01") + c(0:183), times = 3),
  # Based on Dirk Eddelbuettel: https://stackoverflow.com/a/21502386
  shelter = c(rep(x = "Shelter 1", times = 184),
             rep(x = "Shelter 2", times = 184),
             rep(x = "Shelter 3", times = 184)),
  number_occupied =
  rpois(n = 184*3,
        lambda = 30) # Draw 552 times from the Poisson distribution
```

```

)
head(simulated_occupancy_data)
#> # A tibble: 6 x 3
#>   date      shelter  number_occupied
#>   <date>    <chr>        <int>
#> 1 2021-07-01 Shelter 1            28
#> 2 2021-07-02 Shelter 1            29
#> 3 2021-07-03 Shelter 1            35
#> 4 2021-07-04 Shelter 1            25
#> 5 2021-07-05 Shelter 1            21
#> 6 2021-07-06 Shelter 1            30

```

In this simulation we first create a list of all the dates in 2021. We repeat that list three times. We assume data for three shelters for every day of the year. To simulate the number of beds that are occupied each night, we draw from a Poisson distribution, assuming a mean number of 30 beds occupied per shelter.

2.3.3 Acquire

We use data made available about Toronto homeless shelters by the City of Toronto. The premise of the data is that each night at 4am a count is made of the occupied beds. To access the data, we use `opendatatoronto` (Gelfand, 2020) and then save our own copy.

```

##### Acquire #####
# Based on code from:
# https://open.toronto.ca/dataset/daily-shelter-overnight-service-occupancy-capacity/
# Thank you to Heath Priston for assistance
toronto_shelters <-
  # Each package is associated with a unique id which can be found in
  # 'For Developers':
  # https://open.toronto.ca/dataset/daily-shelter-overnight-service-occupancy-capacity/
  list_package_resources("21c83b32-d5a8-4106-a54f-010dbe49f6f2") |>
  # Within that package, we are interested in the 2021 dataset
  filter(name == "daily-shelter-overnight-service-occupancy-capacity-2021") |>
  # Having reduce the dataset down to one row we can get the resource
  get_resource()

write_csv(
  x = toronto_shelters,
  file = "toronto_shelters.csv"

```

```

)
head(toronto_shelters)

#> # A tibble: 6 x 32
#>   `_id` OCCUPANCY_DATE ORGANIZATION_ID ORGANIZATION_NAME
#>   <dbl> <date>          <dbl> <chr>
#> 1 7272806 2021-01-01          24 COSTI Immigrant Se~
#> 2 7272807 2021-01-01          24 COSTI Immigrant Se~
#> 3 7272808 2021-01-01          24 COSTI Immigrant Se~
#> 4 7272809 2021-01-01          24 COSTI Immigrant Se~
#> 5 7272810 2021-01-01          24 COSTI Immigrant Se~
#> 6 7272811 2021-01-01          24 COSTI Immigrant Se~
#> # ... with 28 more variables: SHELTER_ID <dbl>,
#> #   SHELTER_GROUP <chr>, LOCATION_ID <dbl>,
#> #   LOCATION_NAME <chr>, LOCATION_ADDRESS <chr>,
#> #   LOCATION_POSTAL_CODE <chr>, LOCATION_CITY <chr>,
#> #   LOCATION_PROVINCE <chr>, PROGRAM_ID <dbl>,
#> #   PROGRAM_NAME <chr>, SECTOR <chr>, PROGRAM_MODEL <chr>,
#> #   OVERNIGHT_SERVICE_TYPE <chr>, PROGRAM_AREA <chr>, ...

```

Not much needs to be done to this to make it similar to the dataset that we were interested in (Figure 2.5). We need to change the names to make them easier to type using `clean_names()`, reduce the columns to only those that are relevant using `select()`, and only keep the second half of the year using `filter()`.

```

toronto_shelters_clean <-
  clean_names(toronto_shelters) |>
  select(occupancy_date, id, occupied_beds) |>
  filter(occupancy_date >= as_date("2021-07-01"))

head(toronto_shelters_clean)
#> # A tibble: 6 x 3
#>   occupancy_date      id occupied_beds
#>   <date>           <dbl>        <dbl>
#> 1 2021-12-27    7323151         50
#> 2 2021-12-27    7323152         18
#> 3 2021-12-27    7323153         28
#> 4 2021-12-27    7323154         50
#> 5 2021-12-27    7323155        NA
#> 6 2021-12-27    7323156        NA

```

All that remains is to save the cleaned dataset.

```
write_csv(
  x = toronto_shelters_clean,
  file = "cleaned_toronto_shelters.csv"
)
```

2.3.4 Explore

First, we load the dataset that we just created.

```
#### Explore ####
toronto_shelters_clean <-
  read_csv(
    "cleaned_toronto_shelters.csv",
    show_col_types = FALSE
)
```

The dataset is on a daily basis for each shelter. We are interested in understanding average usage for each month. To do this, we need to add and add a month column, which we do using `month()` from `lubridate` (Grolemund and Wickham, 2011). By default, `month()` provides the number of the month, and so we include two arguments ‘label’ and ‘abbr’ to get the full name of the month. We remove rows that do not have any data for the number of beds using `drop_na()` from `tidyR`. And we then create a summary statistic on the basis of monthly groups, using `summarize()` from `dplyr` (Wickham et al., 2020a). We use `kable()` from `knitr` (Xie, 2021) to create a table.

```
# Based on code from Florence Vallée-Dubois and Lisa Lendway
toronto_shelters_clean |>
  mutate(occupancy_month = month(occupancy_date,
                                 label = TRUE,
                                 abbr = FALSE)) |>
  drop_na(occupied_beds) |> # We only want rows that have data
  group_by(occupancy_month) |> # We want to know the occupancy by month
  summarize(number_occupied = mean(occupied_beds)) |>
  kable()
```

occupancy_month	number_occupied
July	29.67137
August	30.83975
September	31.65405
October	32.32991
November	33.26980
December	33.57806

TABLE 2.1: Homeless shelter usage in Toronto in 2021

Month	Average daily number of occupied beds
July	29.7
August	30.8
September	31.7
October	32.3
November	33.3
December	33.6

As with before, this looks fine, and achieves what we set out to do. But we can make some tweaks to the defaults to make it look even better (Table 2.1). We can add a caption, make the column names easier to read, only show an appropriate level of decimal places, and improve the formatting.

```
toronto_shelters_clean |>
  mutate(occupancy_month = month(occupancy_date,
                                 label = TRUE,
                                 abbr = FALSE)) |>
  drop_na(occupied_beds) |> # We only want rows that have data
  group_by(occupancy_month) |> # We want to know the occupancy by month
  summarize(number_occupied = mean(occupied_beds)) |>
  kable(caption = "Homeless shelter usage in Toronto in 2021",
        col.names = c("Month", "Average daily number of occupied beds"),
        digits = 1,
        booktabs = TRUE,
        linesep = "")
)
```

2.3.5 Communicate

We need to write a few brief paragraphs about what we did, why we did it, and what we found. An example follows.

Toronto has a large homeless population. Freezing winters mean it is critical there are enough places in shelters. We are interested to understand how usage of shelters changes in colder months, compared with warmer months.

We use data provided by the City of Toronto about Toronto homeless shelter bed occupancy. Specifically, at 4am each night

a count is made of the occupied beds. We are interested in averaging this over the month. We cleaned, tidied, and analyzed the dataset using the statistical programming language R ([R Core Team, 2021](#)) as well as the packages `tidyverse` ([Wickham, 2017](#)), `janitor` ([Firke, 2020](#)), `tidyr` ([Wickham, 2021b](#)), `opendatatoronto` ([Gelfand, 2020](#)), `lubridate` ([Grolemund and Wickham, 2011](#)), and `knitr` ([Xie, 2021](#)). We then made a table of the average number of occupied beds each night for each month (Table 2.1).

We found that the daily average number of occupied beds was higher in December 2021 than July 2021, with 34 occupied beds in December, compared with 30 in July (Table 2.1). More generally, there was a steady increase in the daily average number of occupied beds between July and December, with a slight increase each month.

The dataset is on the basis of shelters, and so our results may be skewed by changes that are specific to especially large or especially small shelters. It may be that particular shelters are especially attractive in colder months. Additionally, we were concerned with counts of the number of occupied beds, but if the supply of beds changes over the season, then an additional statistic of interest would be proportion occupied.

Although this example is only a few paragraphs, it could be reduced to form an abstract, or increased to form a full report. The first paragraph is a general overview, the second focuses on the data, the third on the results, and the fourth is a discussion. Each of these could be expanded to form sections of a short report.

2.4 Neonatal mortality

Neonatal mortality refers to a death that occurs within the first month of life, and in particular, the neonatal mortality rate (NMR) is the number of neonatal deaths per 1,000 live births ([UN IGME, 2021](#)). Reducing it is part of the third Sustainable Development Goal ([Hug et al., 2019](#)). In this example we will create a graph of the estimated NMR for the past fifty years for: Argentina, Australia, Canada, and Kenya.

2.4.1 Plan

For this example, we need to think about what our dataset should look like, and what the graph should look like.

The dataset needs to have columns that specify the country, and the year. It also needs to have a column with the NMR estimate for that year for that country. Roughly, it should look like Figure 2.7.

country	year	nmr
Argentina	1971	50.0
:	1972	48.7
	:	:
Australia	1971	10.3
:	:	:

FIGURE 2.7: Quick sketch of a potentially useful NMR dataset

We are interested to make a graph with year on the x-axis and estimated NMR on the y-axis. Each country should have its own series Roughly similar to Figure 2.8.

2.4.2 Simulate

We would like to simulate some data that aligns with our plan. In this case we will need three columns: country, year, and NMR.

Within R Studio Cloud make a new R Markdown file and save it. Add preamble documentation and set-up the workspace. We will use `tidyverse` (Wickham, 2017), `janitor` (Firke, 2020), and `lubridate` (Grolemund and Wickham, 2011).

```
#### Preamble ####
# Purpose: Obtain and prepare data about neonatal mortality for four
# countries for the past fifty years and create a graph.
# Author: Rohan Alexander
# Email: rohan.alexander@utoronto.ca
# Date: 1 January 2022
# Prerequisites: -
```

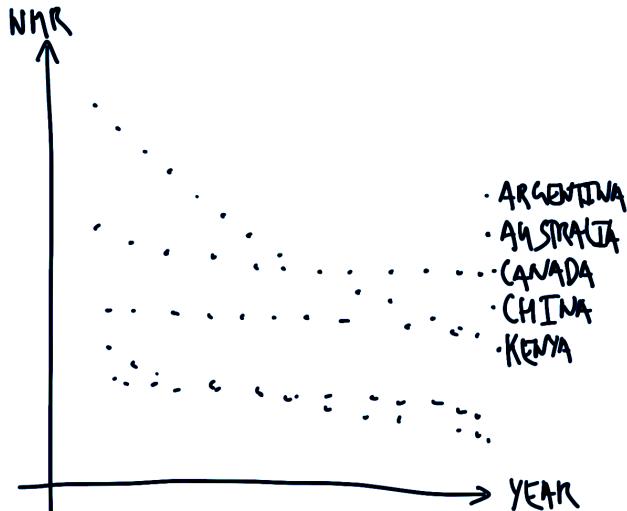


FIGURE 2.8: Quick sketch of a graph of NMR by country over time

```
#### Workspace set-up ####
library(janitor)
library(lubridate)
library(tidyverse)
```

The code contained in libraries can change from time to time as the authors update it and release new versions. We can see which version of a package we are using with `packageVersion()`. For instance, we are using version 1.3.1 of the `tidyverse` and version 2.1.0 of `janitor`.

```
packageVersion('tidyverse')
#> [1] '1.3.1'
packageVersion('janitor')
#> [1] '2.1.0'
```

To update the version of the package, we use `update.packages()`.

```
update.packages()
```

This does not need to be run, say, every day, but from time-to-time it is worth updating packages. While many packages take care to ensure backward

compatibility, at a certain point this does not become reasonable, and so it is important to be aware the updating packages can result in old code needing to be updated.

Returning to the simulation, we repeat the name of each country 50 times with `rep()`, and enable the passing of 50 years. Finally, we draw from the uniform distribution with `runif()` to simulate an estimated NMR value for that year for that country.

```
#### Simulate data ####
set.seed(853)

simulated_nmr_data <-
  tibble(
    country =
      c(
        rep('Argentina', 50),
        rep('Australia', 50),
        rep('Canada', 50),
        rep('Kenya', 50)
      ),
    year =
      rep(c(1971:2020), 4),
    nmr =
      runif(n = 200,
            min = 0,
            max = 100)
  )

head(simulated_nmr_data)
#> # A tibble: 6 x 3
#>   country     year     nmr
#>   <chr>     <int> <dbl>
#> 1 Argentina  1971  35.9
#> 2 Argentina  1972  12.0
#> 3 Argentina  1973  48.4
#> 4 Argentina  1974  31.6
#> 5 Argentina  1975  3.74
#> 6 Argentina  1976  40.4
```

While this simulation works, it would be time-consuming and error-prone if we decided that instead of fifty years, we were interested in simulating, say, sixty years. One way to make this easier is to replace all instances of 50 with a variable. An example follows.

```
#### Simulate data ####
set.seed(853)

number_of_years <- 50

simulated_nmr_data <-
  tibble(
    country =
      c(
        rep('Argentina', number_of_years),
        rep('Australia', number_of_years),
        rep('Canada', number_of_years),
        rep('Kenya', number_of_years)
      ),
    year =
      rep(c(1:number_of_years + 1970), 4),
    nmr =
      runif(n = number_of_years * 4,
            min = 0,
            max = 100)
  )

head(simulated_nmr_data)
#> # A tibble: 6 x 3
#>   country     year     nmr
#>   <chr>     <dbl>   <dbl>
#> 1 Argentina  1971  35.9
#> 2 Argentina  1972  12.0
#> 3 Argentina  1973  48.4
#> 4 Argentina  1974  31.6
#> 5 Argentina  1975  3.74
#> 6 Argentina  1976  40.4
```

The result will be the same, but now if we want to change from fifty to sixty years is to make the change in one place.

We can have confidence in this simulated dataset because it is relatively straight-forward, and we wrote the code for it. But when we turn to the real dataset, it is more difficult to be sure that it is what it claims to be. Even if we trust the data, it is important that we can share that confidence with others. One way forward is to establish some checks that prove our data are as they should be. For instance, we expect:

1. That ‘country’ is, and only is, one of these four: ‘Argentina’, ‘Australia’, ‘Canada’, or ‘Kenya’.
2. Conversely, that ‘country’ contains all those four countries.
3. That ‘year’ is no smaller than 1971 and no larger than 2020, and is a number, not a letter.
4. That ‘nmr’ is a value somewhere between 0 and 1,000, and is a number.

We can write a series of tests based on these features, that we expect that dataset to pass.

```
# Tests for simulated data
simulated_nmr_data$country |>
  unique() == c("Argentina",
              "Australia",
              "Canada",
              "Kenya")
#> [1] TRUE TRUE TRUE TRUE

simulated_nmr_data$country |> unique() |> length() == 4
#> [1] TRUE

simulated_nmr_data$year |> min() == 1971
#> [1] TRUE

simulated_nmr_data$year |> max() == 2020
#> [1] TRUE

simulated_nmr_data$nmr |> min() >= 0
#> [1] TRUE

simulated_nmr_data$nmr |> max() <= 1000
#> [1] TRUE

simulated_nmr_data$nmr |> class() == "numeric"
#> [1] TRUE
```

Having passed these tests, we can have confidence in the simulated dataset. More importantly, we can apply these tests to the real dataset. This enables us to have greater confidence in that dataset and to share that confidence with others.

2.4.3 Acquire

The UN Inter-agency Group for Child Mortality Estimation (IGME) provides estimates of the NMR – <https://childmortality.org/> – that we can download and save.

```
##### Acquire data #####
raw_igme_data <-
  read_csv(
    file =
      "https://childmortality.org/wp-content/uploads/2021/09/UNIGME-2021.csv",
    show_col_types = FALSE)

write_csv(
  x = raw_igme_data,
  file = "igme.csv"
)
```

We can take a quick look to get a better sense of it. We might be interested in what the dataset seems to look like (using `head()` and `tail()`), and what the names of the columns are (using `names()`).

```
head(raw_igme_data)
#> # A tibble: 6 x 29
#>   `Geographic area` `Indicator`       Sex   `Wealth Quintil~
#>   <chr>            <chr>          <chr> <chr>
#> 1 Afghanistan     Neonatal mortali~ Total  Total
#> 2 Afghanistan     Neonatal mortali~ Total  Total
#> 3 Afghanistan     Neonatal mortali~ Total  Total
#> 4 Afghanistan     Neonatal mortali~ Total  Total
#> 5 Afghanistan     Neonatal mortali~ Total  Total
#> 6 Afghanistan     Neonatal mortali~ Total  Total
#> # ... with 25 more variables: Series Name <chr>,
#> #   Series Year <chr>, Regional group <chr>,
#> #   TIME_PERIOD <chr>, OBS_VALUE <dbl>,
#> #   COUNTRY_NOTES <chr>, CONNECTION <lgl>,
#> #   DEATH_CATEGORY <lgl>, CATEGORY <chr>,
#> #   Observation Status <chr>, Unit of measure <chr>,
#> #   Series Category <chr>, Series Type <chr>, ...
names(raw_igme_data)
#> [1] "Geographic area"        "Indicator"
#> [3] "Sex"                   "Wealth Quintile"
#> [5] "Series Name"           "Series Year"
#> [7] "Regional group"         "TIME_PERIOD"
```

```
#> [9] "OBS_VALUE"           "COUNTRY_NOTES"
#> [11] "CONNECTION"        "DEATH_CATEGORY"
#> [13] "CATEGORY"          "Observation Status"
#> [15] "Unit of measure"   "Series Category"
#> [17] "Series Type"       "STD_ERR"
#> [19] "REF_DATE"          "Age Group of Women"
#> [21] "Time Since First Birth" "DEFINITION"
#> [23] "INTERVAL"          "Series Method"
#> [25] "LOWER_BOUND"       "UPPER_BOUND"
#> [27] "STATUS"             "YEAR_TO_ACHIEVE"
#> [29] "Model Used"
```

We would like to clean up the names and only keep the rows and columns that we are interested in. Based on our plan, we are interested in rows where ‘Sex’ is ‘Total’, ‘Series Name’ is ‘UN IGME estimate’, ‘Geographic area’ is one of ‘Argentina’, ‘Australia’, ‘Canada’, and ‘Kenya’, and the ‘Indicator’ is ‘Neonatal mortality rate’. After this we are interested in just a few columns: ‘geographic_area’, ‘time_period’, and ‘obs_value’.

```
cleaned_igme_data <-  
  clean_names(raw_igme_data) |>  
  filter(sex == 'Total',  
         series_name == 'UN IGME estimate',  
         geographic_area %in%  
           c('Argentina', 'Australia', 'Canada', 'Kenya'),  
         indicator == 'Neonatal mortality rate') |>  
  select(geographic_area,  
         time_period,  
         obs_value)  
  
head(cleaned_igme_data)  
#> # A tibble: 6 x 3  
#>   geographic_area time_period obs_value  
#>   <chr>          <chr>        <dbl>  
#> 1 Argentina      1970-06     24.9  
#> 2 Argentina      1971-06     24.7  
#> 3 Argentina      1972-06     24.6  
#> 4 Argentina      1973-06     24.6  
#> 5 Argentina      1974-06     24.5  
#> 6 Argentina      1975-06     24.1
```

Finally, we need to fix two final aspects: the class of ‘time_period’ is character when we need it to be a year, and the name of ‘obs_value’ should be ‘nmr’ to be more informative.

```

cleaned_igme_data <-
  cleaned_igme_data |>
  mutate(time_period = str_remove(time_period, "-06"),
         time_period = as.integer(time_period)) |>
  filter(time_period >= 1971) |>
  rename(nmr = obs_value,
         year = time_period,
         country = geographic_area)

head(cleaned_igme_data)
#> # A tibble: 6 x 3
#>   country     year    nmr
#>   <chr>     <int> <dbl>
#> 1 Argentina  1971  24.7
#> 2 Argentina  1972  24.6
#> 3 Argentina  1973  24.6
#> 4 Argentina  1974  24.5
#> 5 Argentina  1975  24.1
#> 6 Argentina  1976  23.3

```

Finally, we can check that our dataset passes the tests that we developed based on the simulated dataset.

```

# Test the cleaned dataset
cleaned_igme_data$country |>
  unique() == c("Argentina",
              "Australia",
              "Canada",
              "Kenya")
#> [1] TRUE TRUE TRUE TRUE

cleaned_igme_data$country |> unique() |> length() == 4
#> [1] TRUE

cleaned_igme_data$year |> min() == 1971
#> [1] TRUE

cleaned_igme_data$year |> max() == 2020
#> [1] TRUE

cleaned_igme_data$nmr |> min() >= 0
#> [1] TRUE

```

```
cleaned_igme_data$nmr |> max() <= 1000
#> [1] TRUE

cleaned_igme_data$nmr |> class() == "numeric"
#> [1] TRUE
```

All that remains is to save the nicely cleaned dataset.

```
write_csv(
  x = cleaned_igme_data,
  file = "cleaned_igme_data.csv"
)
```

2.4.4 Explore

We would like to make a graph of estimated NMR using the cleaned dataset. First, we read in the dataset.

```
#### Explore ####
cleaned_igme_data <-
  read_csv(
    file = "cleaned_igme_data.csv",
    show_col_types = FALSE
  )
```

We can now make the graph that we are interested in (Figure 2.9). We are interested in showing how NMR has changed over time and the difference between countries.

```
cleaned_igme_data |>
  ggplot(aes(x = year, y = nmr, color = country)) +
  geom_point() +
  theme_minimal() +
  labs(x = "Year",
       y = "Neonatal Mortality Rate (NMR)",
       color = "Country") +
  scale_color_brewer(palette = "Set1")
```

2.4.5 Communicate

To this point we downloaded some data, cleaned it, wrote some tests, and made a graph. We would typically need to communicate what we have done

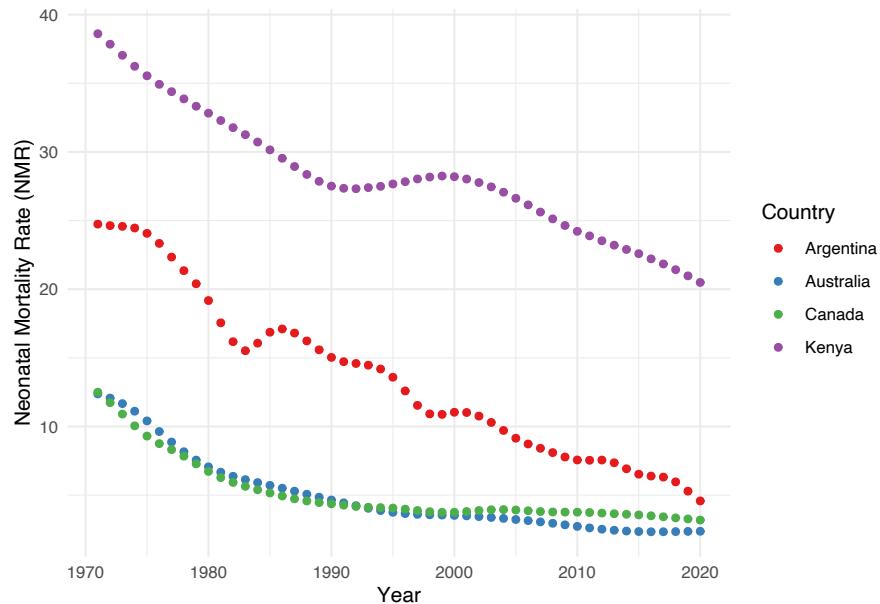


FIGURE 2.9: Neonatal Mortality Rate (NMR), for Argentina, Australia, Canada, and Kenya, (1971-2020)

at some length. In this case, we will write a few paragraphs about what we did, why we did it, and what we found.

Neonatal mortality refers to a death that occurs within the first month of life. In particular, the neonatal mortality rate (NMR) is the number of neonatal deaths per 1,000 live births (Alexander and Alkema, 2018). We obtain estimates for NMR for four countries—Argentina, Australia, Canada, China, and Kenya—over the past fifty years.

The UN Inter-agency Group for Child Mortality Estimation (IGME) provides estimates of the NMR at the website: <https://childmortality.org/>. We downloaded their estimates then cleaned and tidied the dataset using the statistical programming language R (R Core Team, 2021).

We found considerable change in the estimated NMR over time and between the four countries of interest (Figure 2.9). We found that the 1970s tended to be associated with reductions in the estimated NMR. Australia and Canada were estimated to have a

low NMR at that point and remained there through 2020, with slight improvements. The estimates for Argentina and Kenya continued to have substantial reductions through 2020. Data were only available from 1990 for China and the estimates show a substantial reduction in the NMR, especially in the 1990s and 2000s.

Our results suggest considerable improvements in estimated NMR over time. But it is worth emphasizing that estimates of the NMR are based on a statistical model and underlying data. The paradox of data availability is that often high-quality data are less easily available for countries with worse outcomes. For instance, Alexander and Alkema (2018) say ‘[t]here is large variability in the availability of data on neonatal mortality’. Our conclusions are subject to the model that underpins the estimates, and the quality of the underlying data and we did not independently verify either of these.

2.5 Exercises and tutorial

2.5.1 Exercises

1. Following Barrett (2021a), please write a stack of four or five atomic habits that you could implement this week.
2. What is not one of the four challenges for mitigating bias mentioned in Hao (2019) (pick one)?
 - a. Unknown unknowns.
 - b. Imperfect processes.
 - c. The definitions of fairness.
 - d. Lack of social context.
 - e. Disinterest given profit considerations.
3. When was the dataset that underpins Chambliss (1989) collected (pick one)?
 - a. August 1983 to August 1984
 - b. January 1983 to August 1984
 - c. January 1983 to January 1984
 - d. August 1983 to January 1984
4. When Chambliss (1989) talks of stratification, what is he talking about?
5. How does Chambliss (1989) define ‘excellence’ (pick one)?
 - a. Prolonged performance at world-class level.

- b. All Olympic medal winners.
 - c. Consistent superiority of performance.
 - d. All national-level athletes.
6. Think about the following quote from [Chambliss \(1989, p.81\)](#) and list three small skills or activities that could help you achieve excellence in data science.
-

Excellence is mundane. Superlative performance is really a confluence of dozens of small skills or activities, each one learned or stumbled upon, which have been carefully drilled into habit and then are fitted together in a synthesized whole. There is nothing extraordinary or super-human in any one of those actions; only the fact that they are done consistently and correctly, and all together, produce excellence.

7. Which of the following are arguments for `read_csv()` from `readr` ([Wickham et al., 2021](#)) (select all that apply)? (Hint: You can access the help for the function with `?readr::read_csv()`.)
 - a. ‘all_cols’
 - b. ‘file’
 - c. ‘show_col_types’
 - d. ‘number’
8. We used `rpois()` and `runif()` to draw from the Poisson and Uniform distributions, respectively. Which of the following can be used to draw from the Normal and Binomial distributions (select all that apply)?
 - a. `rnormal()` and `rbinom()`
 - b. `rnorm()` and `rbinomial()`
 - c. `rnormal()` and `rbinomial()`
 - d. `rnorm()` and `rbinom()`
9. What is the result of `sample(x = letters, size = 2)` when the seed is set to ‘853’? What about when the seed is set to ‘1234’ (pick one)?
 - a. ‘i’ ‘q’ and ‘p’ ‘v’
 - b. ‘e’ ‘l’ and ‘e’ ‘r’
 - c. ‘i’ ‘q’ and ‘e’ ‘r’
 - d. ‘e’ ‘l’ and ‘p’ ‘v’
10. Which function provides the recommended citation to cite R (pick one)?
 - a. `cite('R')`.

- b. `cite()`.
 - c. `citation('R')`.
 - d. `citation()`.
11. How do we get the citation information for `opendatatoronto` (pick one)?
- a. `cite()`
 - b. `citation()`
 - c. `cite('opendatatoronto')`
 - d. `citation('opendatatoronto')`
12. Which argument needs to be changed to change the headings in `kable()` from `knitr` (Xie, 2021) (pick one)?
- a. ‘booktabs’
 - b. ‘col.names’
 - c. ‘digits’
 - d. ‘linesep’
 - e. ‘caption’
13. Which function is used to update packages (pick one)?
- a. `update.packages()`
 - b. `upgrade.packages()`
 - c. `revise.packages()`
 - d. `renovate.packages()`
14. What are some features that we might typically expect of a column that claimed to be a year (select all that apply)?
- a. The class is ‘character’.
 - b. There are no negative numbers.
 - c. There are letters in the column.
 - d. Each entry has four digits.

2.5.2 Tutorial

1. Please pick either the seed-growing, or hair-length, example from Chapter 1. You should have already started to record some data, but you probably do not have much of it. First sketch an example of what the dataset could look like. Please use `sample()` to create a tibble that has twelve weeks’ worth of simulated data.
2. Pretend that the dataset that you just generated is the actual data that you end up with. Please write a page or so to communicate what you did, why, and what you found.
3. Reflecting on Chambliss (1989), please write a page or so about stratification and excellence as it relates to using programming languages, such as R or Python, for data science.



3

R essentials

Required material

- Read *The Kitchen Counter Observatory*, ([Healy, 2020](#))
- Read *R for Data Science*, Chapter 5 ‘Data transformation’, ([Wickham and Grolemund, 2017](#))
- Read *Data Feminism*, Chapter 6 ‘The Numbers Don’t Speak for Themselves’, ([D’Ignazio and Klein, 2020](#))

Key concepts and skills

- Understanding foundational aspects of R and R Studio.
- Being able to use key `dplyr` verbs.
- Know fundamentals of class and how to manipulate it.
- Ability to simulate data.
- Ability to make graphs in `ggplot2`.
- Comfort with other aspects of the `tidyverse` including importing data, dataset manipulation, string manipulation, and factors.
- Develop strategies for when things do not work.

Key libraries

- `forcats` ([Wickham, 2020a](#))
- `ggplot2` ([Wickham, 2016](#))
- `haven` ([Wickham and Miller, 2020](#))
- `stringr` ([Wickham, 2019e](#))
- `tidyverse` ([Wickham, 2021b](#))
- `tidyverse` ([Wickham et al., 2019a](#))

Key functions

- `|` ‘or’
- `&` ‘and’
- `|>` ‘pipe’
- `$` ‘extract’
- `as.character()`
- `as.integer()`
- `c()`
- `citation()`
- `class()`

- dplyr::arrange()
- dplyr::case_when()
- dplyr::count()
- dplyr::filter()
- dplyr::group_by()
- dplyr::if_else()
- dplyr::left_join()
- dplyr::mutate()
- dplyr::pull()
- dplyr::rename()
- dplyr::select()
- dplyr::slice()
- dplyr::summarise()
- forcats::as_factor()
- forcats::fct_relevel()
- function()
- ggplot2::facet_wrap()
- ggplot2::geom_density()
- ggplot2::geom_histogram()
- ggplot2::geom_point()
- ggplot2::ggplot()
- head()
- janitor::clean_names()
- library()
- lubridate::ymd()
- max()
- mean()
- print()
- readr::read_csv()
- rnorm()
- round()
- runif()
- sample()
- set.seed()
- stringr::str_detect()
- stringr::str_replace()
- stringr::str_squish()
- sum()
- tibble::tibble()
- tidyverse::pivot_longer()
- tidyverse::pivot_wider()

3.1 Background

In this chapter we focus on foundational skills needed to use the statistical programming language R (R Core Team, 2021) to tell stories with data. Some of it may not make sense at first, but these are skills and approaches that we will often use. You should initially just go through this chapter quickly, noting aspects that you do not understand. And then come back to this chapter from time to time as you continue through the rest of the book. That way you will see how the various bits fit into context.

R is an open-source language for statistical programming. You can download R for free from the Comprehensive R Archive Network (CRAN): <https://cran.r-project.org>. R Studio is an Integrated Development Environment (IDE) for R which makes the language easier to use and can be downloaded for free: <https://www.rstudio.com/products/rstudio/>.

The past ten years or so, have been characterized by the increased use of the `tidyverse`. This is ‘...an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures’ (Wickham, 2020b). There are three distinctions to be clear about: the original R language, typically referred to as ‘base’; the ‘tidyverse’ which is a coherent collection of packages that build on top of base, and other packages.

Essentially everything that we can do in the tidyverse, we can also do in base. But, as the `tidyverse` was built especially for data science it is often easier to use the `tidyverse`, especially when learning. Additionally, often everything that we can do in the `tidyverse`, we can also do with other packages. But, as the `tidyverse` is a coherent collection of packages, it is often easier to use the `tidyverse`, again, especially when learning. Eventually there are cases where it makes sense to trade-off the convenience and coherence of the `tidyverse` for some features of base or other packages. Indeed, we will see that at various points later in this book. For instance, the `tidyverse` can be slow, and so if one needs to import thousands of CSVs then it can make sense to switch away from `read_csv()`. The appropriate use of base and non-tidyverse packages, or even other languages, rather than dogmatic insistence on a particular solution, is a sign of intellectual maturity.

Central to our use of the statistical programming language R is data, and most of the data that we use will have humans at the heart of it. Sometimes, dealing with human-centered data in this way can have a numbing effect, results in over-generalization, and potentially problematic work. Another sign of intellectual maturity is when it has the opposite effect.

In practice, I find that far from distancing you from questions of meaning, quantitative data forces you to confront them. The numbers draw you in. Working with data like this is an unending exercise in humility, a constant compulsion to think through what you can and cannot see, and a standing invitation to understand what the measures really capture—what they mean, and for whom.

Healy (2020)

3.2 Broader impacts

“We shouldn’t have to think about the societal impact of our work because it’s hard and other people can do it for us” is a really bad argument. I stopped doing CV [computer vision] research because I saw the impact my work was having. I loved the work but the military applications and privacy concerns eventually became impossible to ignore. But basically all facial recognition work would not get published if we took Broader Impacts sections seriously. There is almost no upside and enormous downside risk. To be fair though i should have a lot of humility here. For most of grad school I bought in to the myth that science is apolitical and research is objectively moral and good no matter what the subject is.

Joe Redmon, 20 February 2020

Although the term ‘data science’ is ubiquitous in academia, industry, and even more generally, it is difficult to define. One deliberately antagonistic definition of data science is ‘[t]he inhumane reduction of humanity down to what can be counted’ (Keyes, 2019). While purposefully controversial, this definition highlights one reason for the increased demand for data science and quantitative methods over the past decade—individuals and their behavior are now at the heart of it. Many of the techniques have been around for many decades, but what makes them popular now is this human focus.

Unfortunately, even though much of the work may be focused on individuals, issues of privacy and consent, and ethical concerns more broadly, rarely seem front of mind. While there are some exceptions, in general, even at the same time as claiming that AI, machine learning, and data science are going to revolutionize society, consideration of these types of issues appears to have been largely treated as something that would be nice to have, rather than something that we may like to think of before we embrace the revolution.

For the most part, these types of issues are not new. In the sciences, there has been considerable recent ethical consideration around CRISPR technology and gene editing ([Brokowski and Adli, 2019](#)), but in an earlier time similar conversations were had, for instance, about Wernher von Braun being allowed to build rockets for the US ([Neufeld, 2002](#)). In medicine, of course, these concerns have been front-of-mind for some time ([Association of Medicine, 1848](#)). Data science seems determined to have its own Tuskegee-moment rather than think about, and proactively deal appropriately with, these issues, based on the experiences of other fields.

That said, there is some evidence that data scientists are beginning to be more concerned about the ethics surrounding the practice. For instance, NeurIPS, a prestigious machine learning conference, has required a statement on ethics to accompany all submissions since 2020.

In order to provide a balanced perspective, authors are required to include a statement of the potential broader impact of their work, including its ethical aspects and future societal consequences. Authors should take care to discuss both positive and negative outcomes.

NeurIPS 2020 Conference Call For Papers

The purpose of ethical consideration and concern for the broader impact of data science is not to prescriptively rule things in or out, but to provide an opportunity to raise some issues that should be paramount. The variety of data science applications, the relative youth of the field, and the speed of change, mean that such considerations are sometimes knowingly set aside, and this is acceptable to the rest of the field. This contrasts with fields such as science, medicine, engineering, and accounting. Possibly those fields are more self-aware (Figure 3.1).

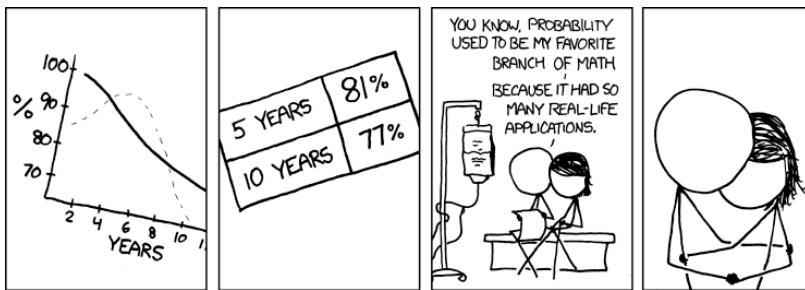


FIGURE 3.1: Probability, from XKCD

3.3 R, R Studio, and R Studio Cloud

R and R Studio are complementary, but they are not the same thing. Liza Bolton, Assistant Professor, Teaching Stream, University of Toronto explains their relationship by analogy where R is like the engine and R Studio is like the car. Although some of us use a car engine directly, most of us use a car to interact with the engine.

3.3.1 R

R – <https://www.r-project.org/> – is an open-source and free programming language that is focused on general statistics. Free in this context does not refer to a price of zero, but instead to the freedom that the creators give users to largely do what they want with it, although it also does have a price of zero. This is in contrast with an open-source programming language that is designed for general purpose, such as Python, or an open-source programming language that is focused on probability, such as Stan. It was created by Ross Ihaka and Robert Gentleman at the University of Auckland in the 1990s, and traces its provenance to S, which was developed at Bell Labs in the 1970s. It is maintained by the R Core Team and changes to this ‘base’ of code occur methodically and with concern given to a variety of different priorities.

Many people build on this stable base, to extend the capabilities of R to better and more quickly suit their needs. They do this by creating packages. Typically, although not always, a package is a collection of R code, mostly functions, and this allows us to more easily do things that we want to do. These packages are managed by repositories such as CRAN and Bioconductor.

If you want to use a package then you first need to install it on your computer, and then you need to load it when you want to use it. Di Cook, Professor of Business Analytics at Monash University, describes this as analogous to a lightbulb. If you want light in your house, first you need to fit a light-

bulb, and then you need to turn the switch on. Installing a package, say, `install.packages("tidyverse")`, is akin to fitting a lightbulb into a socket—you only need to do this once for each lightbulb. But then each time you want light you need to turn on the switch to the lightbulb, which in the R packages case, means calling the library, say, `library(tidyverse)`.

Shoulders of giants Dr Di Cook is Professor of Business Analytics at Monash University. After taking a PhD in statistics from Rutgers University in 1993 where she focused on statistical graphics, she was appointed as an assistant professor at Iowa State University, being promoted to full professor in 2005, and in 2015 she moved to Monash. One area of her research is data visualisation, especially interactive and dynamic graphics. One particularly important paper is Buja et al. (1996) which proposes a taxonomy of interactive data visualization and associated software XGobi.

To install a package on your computer (again, we will need to do this only once per computer) we use `install.packages()`.

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

And then when we want to use the package, we use `library()`.

```
library(tidyverse)
```

Having downloaded it, we can open R and use it directly. It is primarily designed to be interacted with through the command line. While this is functional, it can be useful to have a richer environment than the command line provides. In particular, it can be useful to install an Integrated Development Environment (IDE), which is an application that brings together various bits and pieces that will be used often. One common IDE for R is R Studio, although others such as Visual Studio are also used.

3.3.2 R Studio

R Studio is distinct to R, and they are different entities. R Studio builds on top of R to make it easier to use R. This is in the same way that one could

use the internet from the command line, but most folks use a browser such as Chrome, Firefox, or Safari.

R Studio is free in the sense that we do not pay for it. It is also free in the sense of being able to take the code, modify it, and distribute that code. But it is important to recognize that R Studio is a company and so it is possible that the current situation could change. It can be downloaded: <https://www.rstudio.com/products/rstudio/>.

When we open R Studio it will look like Figure 3.2.

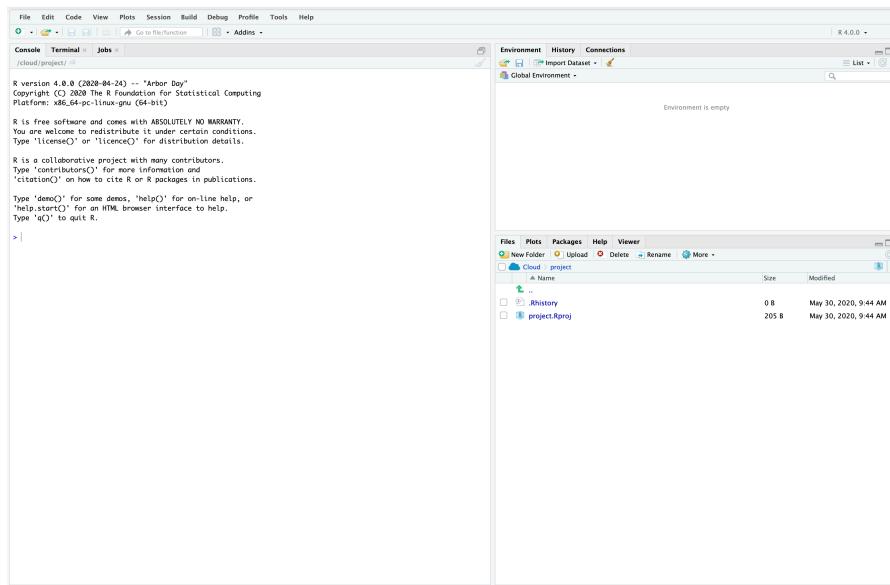


FIGURE 3.2: Opening R Studio for the first time

The left pane is a console in which you can type and execute R code line by line. Try it with $2+2$ by clicking next to the prompt ‘>’, typing ‘ $2+2$ ’, and then pressing ‘return/enter’.

```
2 + 2
#> [1] 4
```

The pane on the top right has information about your environment. For instance, when we create variables a list of their names and some properties will appear there. Try to type the following code, replacing my name with your name, next to the prompt, and again press enter:

```
my_name <- "Rohan"
```

You should notice a new value in the environment pane with the variable name and its value.

The pane in the bottom right is a file manager. At the moment it should just have two files: an R History file and a R Project file. We will get to what these are later, but for now we will create and save a file.

Run the following code, without worrying too much about the details for now. And you should see a new ‘.rds’ file in your list of files.

```
saveRDS(object = my_name, file = "my_first_file.rds")
```

3.3.3 R Studio Cloud

While you can and should download R Studio to your own computer, initially we will use R Studio Cloud: <https://rstudio.cloud/>. This is an online version that is provided by R Studio. We will use this so that you can focus on getting comfortable with R and R Studio in an environment that is consistent. This way you do not have to worry about what computer you have or installation permissions, amongst other things.

The free version of R Studio Cloud is free as is ‘no financial cost’. The trade-off is that it is not very powerful, and it is sometimes slow, but for the purposes of getting started it is enough.

3.4 Getting started

We will now start going through some code. It is important to actively write this all out yourself.

While working line-by-line in the console is fine, it is easier to write out a whole script that can then be run. We will do this by making an R Script (‘File’ -> ‘New File’ -> ‘R Script’). The console pane will fall to the bottom left and an R Script will open in the top left. We will write some code that will get all of the Australian federal politicians and then construct a small table about the genders of the prime ministers. Some of this code will not make sense at this stage, but just type it all out to get in the habit and then run it. To run the whole script, we can click ‘Run’ or we can highlight certain lines and then click ‘Run’ to just run those lines.

```
# Install the packages that we need
install.packages("tidyverse")
install.packages("AustralianPoliticians")

# Load the packages that we need to use this time
library(tidyverse)
library(AustralianPoliticians)

# Make a table of the counts of genders of the prime ministers
AustralianPoliticians::get_auspol('all') |>
  as_tibble() |>
  filter(wasPrimeMinister == 1) |>
  count(gender)
#> # A tibble: 2 x 2
#>   gender     n
#>   <chr>   <int>
#> 1 female      1
#> 2 male       29
```

We can see that, as at the end of 2021, one female has been prime minister (Julia Gillard), while the other 29 prime ministers were male

One critical operator when programming is the ‘pipe’: `|>`. We read this as ‘and then’. This takes the output of a line of code and uses it as the first input to the next line of code. It makes code easier to read.

The idea of the pipe is that we take a dataset, and then do something to it. We used this in the earlier example. Another example follows where we will look at the first six lines of a dataset by piping it to `head()`. Notice that `head()` does not explicitly take any arguments in this example. It knows which data to display because the pipe does it implicitly.

```
AustralianPoliticians::get_auspol('all') |>
  head()
#> # A tibble: 6 x 20
#>   uniqueID  surname allOtherNames      firstName commonName
#>   <chr>      <chr>    <chr>      <chr>      <chr>
#> 1 Abbott1859 Abbott  Richard Hartley S~ Richard    <NA>
#> 2 Abbott1869 Abbott  Percy Phipps      Percy      <NA>
#> 3 Abbott1877 Abbott  Macartney        Macartney  Mac
#> 4 Abbott1886 Abbott  Charles Lydiard A~ Charles    Aubrey
#> 5 Abbott1891 Abbott  Joseph Palmer    Joseph    <NA>
#> 6 Abbott1957 Abbott  Anthony John    Anthony   Tony
#> # ... with 15 more variables: displayName <chr>,
```

```
#> #   earlierOrLaterNames <chr>, title <chr>, gender <chr>,
#> #   birthDate <date>, birthYear <dbl>, birthPlace <chr>,
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>, wikidataID <chr>,
#> #   wikipedia <chr>, adb <chr>, comments <chr>
```

We can save this R Script as ‘my_first_r_script.R’ (‘File’ -> ‘Save As’). At this point, our workspace should look something like Figure 3.3.

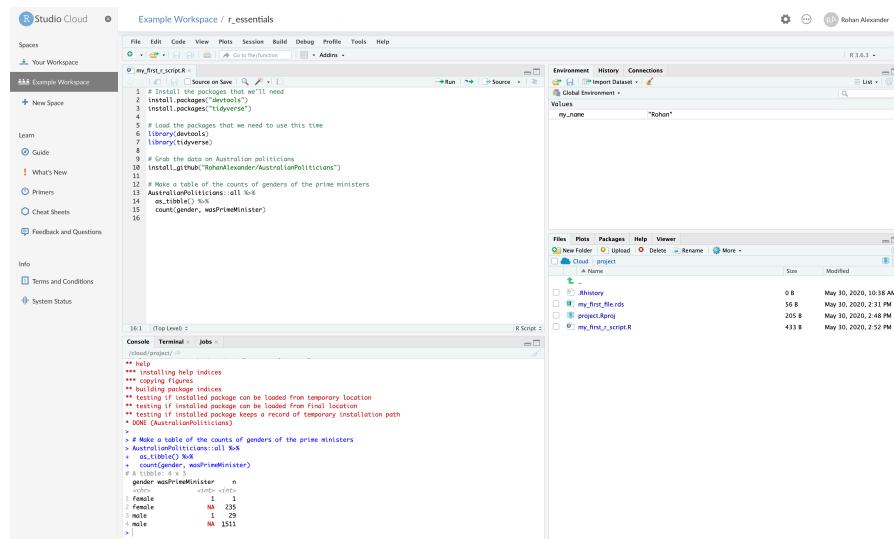


FIGURE 3.3: After running an R Script

One thing to be aware of is that each R Studio Cloud workspace is essentially a new computer. Because of this, we need to install any package that we want to use for each workspace. For instance, before we can use the `tidyverse`, we need to install it with `install.packages("tidyverse")`. This contrasts with using one’s own computer.

A few final notes on R Studio Cloud:

1. In the Australian politician’s example, we got our data from the website GitHub using an R package, but we can get data into a workspace from a local computer in a variety of ways. One way is to use the ‘upload’ button in the ‘Files’ panel.
2. R Studio Cloud allows some degree of collaboration. For instance, you can give someone else access to a workspace that you create. This could be useful for collaborating on an assignment, although

it is not quite full featured yet and you cannot both be in the workspace at the same time, in contrast to, say, Google Docs.

3. There are a variety of weaknesses of R Studio Cloud, in particular the RAM limits. Additionally, like any web application, things break from time to time or go down.

3.5 The `dplyr` verbs

One of the key packages that we will use is the `tidyverse` (Wickham et al., 2019b). The `tidyverse` is actually a package of packages, which means when we install the `tidyverse`, we actually install a whole bunch of different packages. The key package in the `tidyverse` in terms of manipulating data is `dplyr` (Wickham et al., 2020a).

There are five `dplyr` functions that are regularly used, and we will now go through each of these. These are commonly referred to as the `dplyr` verbs.

1. `select()`
2. `filter()`
3. `arrange()`
4. `mutate()`
5. `summarise()` or equally `summarize()`

We will also cover `group_by()`, and `count()` here as they are closely related.

As we have already installed the `tidyverse`, we just need to load it.

```
library(tidyverse)
```

And we will begin by again using some data about Australian politicians from the `AustralianPoliticians` package (Alexander and Hodgetts, 2021).

```
library(AustralianPoliticians)

australian_politicians <-
  get_auspol('all')

head(australian_politicians)
#> # A tibble: 6 x 20
#>   uniqueID surname allOtherNames      firstName commonName
#>   <chr>     <chr>    <chr>          <chr>      <chr>
#> 1 Abbott1859 Abbott  Richard Hartley S~ Richard   <NA>
```

```
#> 2 Abbott1869 Abbott Percy Phipps      Percy      <NA>
#> 3 Abbott1877 Abbott Macartney       Macartney Mac
#> 4 Abbott1886 Abbott Charles Lydiard A~ Charles   Aubrey
#> 5 Abbott1891 Abbott Joseph Palmer    Joseph     <NA>
#> 6 Abbott1957 Abbott Anthony John    Anthony   Tony
#> # ... with 15 more variables: displayName <chr>,
#> #   earlierOrLaterNames <chr>, title <chr>, gender <chr>,
#> #   birthDate <date>, birthYear <dbl>, birthPlace <chr>,
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>, wikidataID <chr>,
#> #   wikipedia <chr>, adb <chr>, comments <chr>
```

3.5.1 `select()`

We use `select()` to pick particular columns of a dataset. For instance, we might like to select the ‘firstName’ column.

```
australian_politicians |>
  select(firstName) |>
  head()

#> # A tibble: 6 x 1
#>   firstName
#>   <chr>
#> 1 Richard
#> 2 Percy
#> 3 Macartney
#> 4 Charles
#> 5 Joseph
#> 6 Anthony
```

In R, there are many ways to do things. Sometimes these are different ways to do the same thing, and other times they are different ways to do almost the same thing. For instance, another way to pick a particular column of a dataset is to use the ‘extract’ operator ‘\$’. This is from base, as opposed to `select()` which is from *tidyverse*.

```
australian_politicians$firstName |>
  head()

#> [1] "Richard"   "Percy"      "Macartney"  "Charles"
#> [5] "Joseph"    "Anthony"
```

The two appear similar—both pick the ‘firstName’ column—but they differ in the class of what they return. For the sake of completeness, if we combine

`select()` with `pull()` then we get the same class of output as if we had used the extract operator.

```
australian_politicians |>
  select(firstName) |>
  pull() |>
  head()
#> [1] "Richard"   "Percy"      "Macartney" "Charles"
#> [5] "Joseph"    "Anthony"
```

We can also use `select()` to remove columns, by negating the column name.

```
australian_politicians |>
  select(-firstName) |>
  head()
#> # A tibble: 6 x 19
#>   uniqueID   surname allOtherNames   commonName displayName
#>   <chr>       <chr>     <chr>           <chr>      <chr>
#> 1 Abbott1859 Abbott  Richard Hartley~ <NA>        Abbott, Ri~
#> 2 Abbott1869 Abbott  Percy Phipps     <NA>        Abbott, Pe~
#> 3 Abbott1877 Abbott  Macartney       Mac          Abbott, Mac
#> 4 Abbott1886 Abbott  Charles Lydiard~ Aubrey    Abbott, Au~
#> 5 Abbott1891 Abbott  Joseph Palmer    <NA>        Abbott, Jo~
#> 6 Abbott1957 Abbott  Anthony John    Tony         Abbott, To~
#> # ... with 14 more variables: earlierOrLaterNames <chr>,
#> #   title <chr>, gender <chr>, birthDate <date>,
#> #   birthYear <dbl>, birthPlace <chr>, deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>,
#> #   wikidataID <chr>, wikipedia <chr>, adb <chr>,
#> #   comments <chr>
```

Finally, we can `select()` based on conditions. For instance, we can `select()` all of the columns that start with, say, ‘birth’.

```
australian_politicians |>
  select(starts_with("birth")) |>
  head()
#> # A tibble: 6 x 3
#>   birthDate   birthYear birthPlace
#>   <date>       <dbl>   <chr>
#> 1 NA            1859   Bendigo
#> 2 1869-05-14    NA    Hobart
#> 3 1877-07-03    NA  Murrurundi
#> 4 1886-01-04    NA St Leonards
```

```
#> 5 1891-10-18      NA North Sydney
#> 6 1957-11-04      NA London
```

There are a variety of similar ‘selection helpers’ including `starts_with()`, `ends_with()`, and `contains()`. More information about these is available in the help page for `select()` which can be accessed by running `?select()`.

At this point, we will use `select()` to reduce the width of our dataset.

```
australian_politicians <-
  australian_politicians |>
  select(uniqueID,
         surname,
         firstName,
         gender,
         birthDate,
         birthYear,
         deathDate,
         member,
         senator,
         wasPrimeMinister)

australian_politicians |> head()
#> # A tibble: 6 x 10
#>   uniqueID   surname firstName gender birthDate   birthYear
#>   <chr>       <chr>    <chr>     <chr>    <date>      <dbl>
#> 1 Abbott1859 Abbott   Richard   male    NA          1859
#> 2 Abbott1869 Abbott   Percy     male    1869-05-14    NA
#> 3 Abbott1877 Abbott   Macartney male    1877-07-03    NA
#> 4 Abbott1886 Abbott   Charles   male    1886-01-04    NA
#> 5 Abbott1891 Abbott   Joseph   male    1891-10-18    NA
#> 6 Abbott1957 Abbott   Anthony  male    1957-11-04    NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

3.5.2 `filter()`

We use `filter()` to pick particular rows of a dataset. For instance, we might be only interested in politicians that became prime minister.

```
australian_politicians |>
  filter(wasPrimeMinister == 1)
#> # A tibble: 30 x 10
```

```
#>   uniqueID    surname firstName gender birthDate birthYear
#>   <chr>      <chr>    <chr>     <chr>   <date>       <dbl>
#> 1 Abbott1957 Abbott Anthony male  1957-11-04     NA
#> 2 Barton1849 Barton Edmund male  1849-01-18     NA
#> 3 Bruce1883 Bruce Stanley male  1883-04-15     NA
#> 4 Chifley1885 Chifley Joseph male 1885-09-22     NA
#> 5 Cook1860   Cook   Joseph male  1860-12-07     NA
#> 6 Curtin1885 Curtin John   male  1885-01-08     NA
#> 7 Deakin1856 Deakin Alfred male  1856-08-03     NA
#> 8 Fadden1894 Fadden Arthur male  1894-04-13     NA
#> 9 Fisher1862 Fisher Andrew male  1862-08-29     NA
#> 10 Forde1890 Forde Francis male 1890-07-18     NA
#> # ... with 20 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

We could also give `filter()` two conditions. For instance, we could look at politicians that become prime minister and were named Joseph, using the ‘and’ operator ‘`&`’.

```
australian_politicians |>
  filter(wasPrimeMinister == 1 & firstName == "Joseph")
#> # A tibble: 3 x 10
#>   uniqueID    surname firstName gender birthDate birthYear
#>   <chr>      <chr>    <chr>     <chr>   <date>       <dbl>
#> 1 Chifley1885 Chifley Joseph male  1885-09-22     NA
#> 2 Cook1860   Cook   Joseph male  1860-12-07     NA
#> 3 Lyons1879  Lyons  Joseph male  1879-09-15     NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

We get the same result if we use a comma instead of an ampersand.

```
australian_politicians |>
  filter(wasPrimeMinister == 1, firstName == "Joseph")
#> # A tibble: 3 x 10
#>   uniqueID    surname firstName gender birthDate birthYear
#>   <chr>      <chr>    <chr>     <chr>   <date>       <dbl>
#> 1 Chifley1885 Chifley Joseph male  1885-09-22     NA
#> 2 Cook1860   Cook   Joseph male  1860-12-07     NA
#> 3 Lyons1879  Lyons  Joseph male  1879-09-15     NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

Similarly, we could look at politicians who were named, say, Myles or Ruth using the ‘or’ operator ‘|’

```
australian_politicians |>
  filter(firstName == "Myles" | firstName == "Ruth")
#> # A tibble: 3 x 10
#>   uniqueID      surname firstName gender birthDate birthYear
#>   <chr>        <chr>    <chr>     <chr>    <date>       <dbl>
#> 1 Coleman1931 Coleman Ruth     female 1931-09-27      NA
#> 2 Ferricks1875 Ferric~ Myles    male   1875-11-12      NA
#> 3 Webber1965  Webber Ruth     female 1965-03-24      NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

We could also pipe the result. For instance we could pipe from `filter()` to `select()`.

```
australian_politicians |>
  filter(firstName == "Ruth" | firstName == "Myles") |>
  select(firstName, surname)
#> # A tibble: 3 x 2
#>   firstName surname
#>   <chr>      <chr>
#> 1 Ruth       Coleman
#> 2 Myles      Ferricks
#> 3 Ruth       Webber
```

If we happen to know the particular row number that is of interest then we could `filter()` to only that particular row. For instance, say the row 853 was of interest.

```
australian_politicians |>
  filter(row_number() == 853)
#> # A tibble: 1 x 10
#>   uniqueID      surname firstName gender birthDate birthYear
#>   <chr>        <chr>    <chr>     <chr>    <date>       <dbl>
#> 1 Jakobsen1947 Jakobs~ Carolyn   female 1947-09-11      NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

There is also a dedicated function to do this, which is `slice()`.

```
australian_politicians |>
  slice(853)
#> # A tibble: 1 x 10
#>   uniqueID      surname firstName gender birthDate birthYear
#>   <chr>        <chr>    <chr>     <chr>    <date>       <dbl>
#> 1 Jakobsen1947 Jakobs~ Carolyn   female  1947-09-11       NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

While this may seem somewhat esoteric, it is especially useful if we would like to remove a particular row using negation, or duplicate specific rows. For instance, we could remove the first row.

```
australian_politicians |>
  slice(-1)
#> # A tibble: 1,782 x 10
#>   uniqueID      surname firstName gender birthDate birthYear
#>   <chr>        <chr>    <chr>     <chr>    <date>       <dbl>
#> 1 Abbott1869 Abbott   Percy     male    1869-05-14       NA
#> 2 Abbott1877 Abbott   Macartney male   1877-07-03       NA
#> 3 Abbott1886 Abbott   Charles   male   1886-01-04       NA
#> 4 Abbott1891 Abbott   Joseph   male   1891-10-18       NA
#> 5 Abbott1957 Abbott   Anthony   male   1957-11-04       NA
#> 6 Abel1939 Abel     John     male   1939-06-25       NA
#> 7 Abetz1958 Abetz    Eric     male   1958-01-25       NA
#> 8 Adams1943 Adams    Judith   female 1943-04-11       NA
#> 9 Adams1951 Adams    Dick     male   1951-04-29       NA
#> 10 Adamson1857 Adamson   John    male   1857-02-18      NA
#> # ... with 1,772 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

We could also only, say, only keep the first three rows.

```
australian_politicians |>
  slice(1:3)
#> # A tibble: 3 x 10
#>   uniqueID      surname firstName gender birthDate birthYear
#>   <chr>        <chr>    <chr>     <chr>    <date>       <dbl>
#> 1 Abbott1859 Abbott   Richard   male    NA          1859
#> 2 Abbott1869 Abbott   Percy     male   1869-05-14       NA
#> 3 Abbott1877 Abbott   Macartney male   1877-07-03       NA
```

```
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

Finally, we could duplicate the first two rows.

```
australian_politicians |>
  slice(1:2, 1:n())
#> # A tibble: 1,785 x 10
#>   uniqueID    surname firstName gender birthDate   birthYear
#>   <chr>        <chr>    <chr>     <chr>   <date>       <dbl>
#> 1 Abbott1859 Abbott   Richard    male    NA          1859
#> 2 Abbott1869 Abbott   Percy      male   1869-05-14      NA
#> 3 Abbott1859 Abbott   Richard    male    NA          1859
#> 4 Abbott1869 Abbott   Percy      male   1869-05-14      NA
#> 5 Abbott1877 Abbott   Macartney  male   1877-07-03      NA
#> 6 Abbott1886 Abbott   Charles    male   1886-01-04      NA
#> 7 Abbott1891 Abbott   Joseph    male   1891-10-18      NA
#> 8 Abbott1957 Abbott   Anthony   male   1957-11-04      NA
#> 9 Abel1939  Abel    John      male   1939-06-25      NA
#> 10 Abetz1958 Abetz   Eric      male   1958-01-25      NA
#> # ... with 1,775 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

3.5.3 `arrange()`

We use `arrange()` to change the order of the dataset based on the values of particular columns. For instance, we could arrange the politicians by their birthday.

```
australian_politicians |>
  arrange(birthDate)
#> # A tibble: 1,783 x 10
#>   uniqueID    surname firstName gender birthDate   birthYear
#>   <chr>        <chr>    <chr>     <chr>   <date>       <dbl>
#> 1 Braddon1829 Braddon   Edward    male   1829-06-11      NA
#> 2 Ferguson18~ Fergus~  John      male   1830-03-15      NA
#> 3 Zeal1830    Zeal     William   male   1830-12-05      NA
#> 4 Fraser1832 Fraser   Simon     male   1832-08-21      NA
#> 5 Groom1833   Groom    William   male   1833-03-09      NA
#> 6 Sargood1834 Sargood  Frederick male   1834-05-30      NA
#> 7 Fysh1835    Fysh     Philip    male   1835-03-01      NA
```

```
#> 8 Playford18~ Playfo~ Thomas    male  1837-11-26      NA
#> 9 Solomon1839 Solomon Elias   male  1839-09-02      NA
#> 10 McLean1840 McLean Allan   male  1840-02-03      NA
#> # ... with 1,773 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

We could modify `arrange()` with `desc()` to change from ascending to descending order.

```
australian_politicians |>
  arrange(desc(birthDate))
#> # A tibble: 1,783 x 10
#>   uniqueID   surname   firstName gender birthDate   birthYear
#>   <chr>       <chr>     <chr>    <chr>  <date>       <dbl>
#> 1 SteeleJoh~ Steele-~ Jordon    male   1994-10-14      NA
#> 2 Chandler1~ Chandler Claire   female  1990-06-01      NA
#> 3 Roy1990    Roy        Wyatt    male   1990-05-22      NA
#> 4 Thompson1~ Thompson Phillip male   1988-05-07      NA
#> 5 Paterson1~ Paterson James   male   1987-11-21      NA
#> 6 Burns1987  Burns       Joshua   male   1987-02-06      NA
#> 7 Smith1986  Smith       Marielle female  1986-12-30      NA
#> 8 Kakoschke~ Kakosch~ Skye    female  1985-12-19      NA
#> 9 Simmonds1~ Simmonds Julian  male   1985-08-29      NA
#> 10 Gorman1984 Gorman      Patrick  male   1984-12-12      NA
#> # ... with 1,773 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

And we could arrange based on more than one column. For instance, if two politicians have the same first name, then we could arrange based on their birthday.

```
australian_politicians |>
  arrange(firstName, birthDate)
#> # A tibble: 1,783 x 10
#>   uniqueID   surname   firstName gender birthDate   birthYear
#>   <chr>       <chr>     <chr>    <chr>  <date>       <dbl>
#> 1 Blain1894  Blain     Adair    male   1894-11-21      NA
#> 2 Dein1889   Dein      Adam    male   1889-03-04      NA
#> 3 Armstrong1~ Armstrong Adam   male   1909-07-01      NA
#> 4 Bandt1972  Bandt     Adam    male   1972-03-11      NA
#> 5 Ridgeway19~ Ridgeway Aden   male   1962-09-18      NA
```

```
#> 6 Bennett1933 Bennett Adrian    male  1933-01-21      NA
#> 7 Gibson1935 Gibson  Adrian    male  1935-11-03      NA
#> 8 Wynne1850  Wynne   Agar     male  1850-01-15      NA
#> 9 Robertson1~ Robert~ Agnes female 1882-07-31      NA
#> 10 Pittard1902 Pittard Alan   male  1902-11-15      NA
#> # ... with 1,773 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

We could achieve the same result by piping between two instances of `arrange()`.

```
australian_politicians |>
  arrange(birthDate) |>
  arrange(firstName)
#> # A tibble: 1,783 x 10
#>   uniqueID   surname firstName gender birthDate birthYear
#>   <chr>       <chr>    <chr>    <chr>   <date>        <dbl>
#> 1 Blain1894  Blain     Adair    male   1894-11-21      NA
#> 2 Dein1889   Dein      Adam     male   1889-03-04      NA
#> 3 Armstrong1~ Armstr~ Adam     male   1909-07-01      NA
#> 4 Bandt1972  Bandt     Adam     male   1972-03-11      NA
#> 5 Ridgeway19~ Ridgew~ Aden    male   1962-09-18      NA
#> 6 Bennett1933 Bennett   Adrian   male   1933-01-21      NA
#> 7 Gibson1935 Gibson   Adrian   male   1935-11-03      NA
#> 8 Wynne1850  Wynne   Agar     male  1850-01-15      NA
#> 9 Robertson1~ Robert~ Agnes female 1882-07-31      NA
#> 10 Pittard1902 Pittard   Alan    male  1902-11-15      NA
#> # ... with 1,773 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

When we use `arrange()` it is important to be clear about precedence. For instance, changing to birthday and then first name would give a different arrangement.

```
australian_politicians |>
  arrange(birthYear, firstName)
#> # A tibble: 1,783 x 10
#>   uniqueID   surname firstName gender birthDate birthYear
#>   <chr>       <chr>    <chr>    <chr>   <date>        <dbl>
#> 1 Edwards1842 Edwards  Richard   male    NA          1842
#> 2 Sawers1844  Sawers  William   male    NA          1844
```

```
#> 3 Barker1846 Barker Stephen male NA 1846
#> 4 Corser1852 Corser Edward male NA 1852
#> 5 Lee1856 Lee Henry male NA 1856
#> 6 Grant1857 Grant John male NA 1857
#> 7 Palmer1859 Palmer Albert male NA 1859
#> 8 Riley1859 Riley Edward male NA 1859
#> 9 Abbott1859 Abbott Richard male NA 1859
#> 10 Kennedy1860 Kennedy Thomas male NA 1860
#> # ... with 1,773 more rows, and 4 more variables:
#> #   deathDate <date>, member <dbl>, senator <dbl>,
#> #   wasPrimeMinister <dbl>
```

A nice way to arrange by a variety of columns is to use `across()`. It enables us to use the ‘selection helpers’ such as `starts_with()` that were mentioned in association with `select()`.

```
australian_politicians |>
  arrange(across(c(firstName, birthYear))) |>
  head()
#> # A tibble: 6 x 10
#>   uniqueID    surname firstName gender birthDate birthYear
#>   <chr>        <chr>     <chr>    <chr>   <date>      <dbl>
#> 1 Blain1894   Blain     Adair    male    1894-11-21     NA
#> 2 Armstrong1~ Armstrong~ Adam    male    1909-07-01     NA
#> 3 Bandt1972   Bandt     Adam    male    1972-03-11     NA
#> 4 Dein1889    Dein      Adam    male    1889-03-04     NA
#> 5 Ridgeway19~ Ridgeway~ Aden    male    1962-09-18     NA
#> 6 Bennett1933 Bennett   Adrian   male    1933-01-21     NA
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>

australian_politicians |>
  arrange(across(starts_with('birth'))) |>
  head()
#> # A tibble: 6 x 10
#>   uniqueID    surname firstName gender birthDate birthYear
#>   <chr>        <chr>     <chr>    <chr>   <date>      <dbl>
#> 1 Braddon1829 Braddon   Edward   male    1829-06-11     NA
#> 2 Ferguson1830 Ferguson~ John    male    1830-03-15     NA
#> 3 Zeal1830    Zeal      William  male    1830-12-05     NA
#> 4 Fraser1832 Fraser   Simon    male    1832-08-21     NA
#> 5 Groom1833   Groom     William  male    1833-03-09     NA
#> 6 Sargood1834 Sargood   Frederick male    1834-05-30     NA
```

```
#> # ... with 4 more variables: deathDate <date>,
#> #   member <dbl>, senator <dbl>, wasPrimeMinister <dbl>
```

3.5.4 `mutate()`

We use `mutate()` when we want to make a new column. For instance, perhaps we want to make a new column that is 1 if a person was both a member and a senator and 0 otherwise. That is to say that our new column would denote politicians that served in both the upper and the lower house.

```
australian_politicians <-
  australian_politicians |>
  mutate(was_both = if_else(member == 1 & senator == 1, 1, 0))

australian_politicians |>
  select(member, senator, was_both)
#> # A tibble: 1,783 x 3
#>   member senator was_both
#>   <dbl>    <dbl>    <dbl>
#> 1     0        1        0
#> 2     1        1        1
#> 3     0        1        0
#> 4     1        0        0
#> 5     1        0        0
#> 6     1        0        0
#> 7     1        0        0
#> 8     0        1        0
#> 9     0        1        0
#> 10    1        0        0
#> # ... with 1,773 more rows
```

We could use `mutate()` with math, such as addition and subtraction. For instance, we could calculate the age that the politicians are (or would have been) in 2022.

```
australian_politicians <-
  australian_politicians |>
  mutate(age = 2022 - lubridate::year(birthDate))

australian_politicians |>
  select(uniqueID, age)
#> # A tibble: 1,783 x 2
```

```
#>   uniqueID      age
#>   <chr>      <dbl>
#> 1 Abbott1859     NA
#> 2 Abbott1869    153
#> 3 Abbott1877    145
#> 4 Abbott1886    136
#> 5 Abbott1891    131
#> 6 Abbott1957     65
#> 7 Abel1939      83
#> 8 Abetz1958     64
#> 9 Adams1943     79
#> 10 Adams1951    71
#> # ... with 1,773 more rows
```

There are a variety of functions that are especially useful when constructing new columns. These include `log()` which will compute the natural logarithm, `lead()` which will bring values up by one row, `lag()` which will push values down by one row, and `cumsum()` which creates a cumulative sum of the column.

```
australian_politicians |>
  select(uniqueID, age) |>
  mutate(log_age = log(age)) |>
  head()

#> # A tibble: 6 x 3
#>   uniqueID      age log_age
#>   <chr>      <dbl>    <dbl>
#> 1 Abbott1859     NA     NA
#> 2 Abbott1869    153    5.03
#> 3 Abbott1877    145    4.98
#> 4 Abbott1886    136    4.91
#> 5 Abbott1891    131    4.88
#> 6 Abbott1957     65    4.17

australian_politicians |>
  select(uniqueID, age) |>
  mutate(lead_age = lead(age)) |>
  head()

#> # A tibble: 6 x 3
#>   uniqueID      age lead_age
#>   <chr>      <dbl>    <dbl>
#> 1 Abbott1859     NA     153
#> 2 Abbott1869    153    145
#> 3 Abbott1877    145    136
```

```
#> 4 Abbott1886    136      131
#> 5 Abbott1891    131      65
#> 6 Abbott1957     65      83

australian_politicians |>
  select(uniqueID, age) |>
  mutate(lag_age = lag(age)) |>
  head()
#> # A tibble: 6 x 3
#>   uniqueID     age  lag_age
#>   <chr>       <dbl>    <dbl>
#> 1 Abbott1859     NA      NA
#> 2 Abbott1869    153      NA
#> 3 Abbott1877    145      153
#> 4 Abbott1886    136      145
#> 5 Abbott1891    131      136
#> 6 Abbott1957     65      131

australian_politicians |>
  select(uniqueID, age) |>
  filter(!is.na(age)) |>
  mutate(cumulative_age = cumsum(age)) |>
  head()
#> # A tibble: 6 x 3
#>   uniqueID     age cumulative_age
#>   <chr>       <dbl>        <dbl>
#> 1 Abbott1869    153          153
#> 2 Abbott1877    145          298
#> 3 Abbott1886    136          434
#> 4 Abbott1891    131          565
#> 5 Abbott1957     65          630
#> 6 Abel1939     83          713
```

As we have in earlier examples, we can also use `mutate()` in combination with `across()`. This includes the potential use of the selection helpers. For instance, we could count the number of characters in both the first and last names at the same time.

```
australian_politicians |>
  mutate(across(c(firstName, surname), str_count)) |>
  select(uniqueID, firstName, surname)
#> # A tibble: 1,783 x 3
#>   uniqueID   firstName   surname
```

```
#>     <chr>      <int>  <int>
#> 1 Abbott1859      7      6
#> 2 Abbott1869      5      6
#> 3 Abbott1877      9      6
#> 4 Abbott1886      7      6
#> 5 Abbott1891      6      6
#> 6 Abbott1957      7      6
#> 7 Abel1939       4      4
#> 8 Abetz1958       4      5
#> 9 Adams1943       6      5
#> 10 Adams1951      4      5
#> # ... with 1,773 more rows
```

Finally, we use `case_when()` when we need to make a new column on the basis of more than two conditional statements. For instance, we may have some years and want to group them into decades.

```
australian_politicians |>
  mutate(year_of_birth = lubridate::year(birthDate),
         decade_of_birth =
           case_when(
             year_of_birth <= 1929 ~ "pre-1930",
             year_of_birth <= 1939 ~ "1930s",
             year_of_birth <= 1949 ~ "1940s",
             year_of_birth <= 1959 ~ "1950s",
             year_of_birth <= 1969 ~ "1960s",
             year_of_birth <= 1979 ~ "1970s",
             year_of_birth <= 1989 ~ "1980s",
             TRUE ~ "Unknown or error"
           )
  ) |>
  select(uniqueID, year_of_birth, decade_of_birth)
#> # A tibble: 1,783 x 3
#>   uniqueID    year_of_birth decade_of_birth
#>   <chr>          <dbl> <chr>
#> 1 Abbott1859      NA Unknown or error
#> 2 Abbott1869      1869 pre-1930
#> 3 Abbott1877      1877 pre-1930
#> 4 Abbott1886      1886 pre-1930
#> 5 Abbott1891      1891 pre-1930
#> 6 Abbott1957      1957 1950s
#> 7 Abel1939       1939 1930s
#> 8 Abetz1958       1958 1950s
```

```
#> 9 Adams1943          1943 1940s
#> 10 Adams1951         1951 1950s
#> # ... with 1,773 more rows
```

We could accomplish this with a series of `if_else()` statements, but `case_when()` is more clear. The cases are evaluated in order and as soon as there is a match `case_when()` does not continue to the remainder of the cases. So it can be useful to have a catch-all at the end that will signal if there is a potential issue that we might like to know about.

3.5.5 `summarise()`

We use `summarise()` when we would like to make new, condensed, summary variables. For instance, perhaps we would like to know the minimum, average, and maximum of some column.

```
australian_politicians |>
  summarise(youngest = min(age, na.rm = TRUE),
             oldest = max(age, na.rm = TRUE),
             average = mean(age, na.rm = TRUE))
#> # A tibble: 1 x 3
#>   youngest  oldest  average
#>       <dbl>    <dbl>    <dbl>
#> 1        28     193     101.
```

As an aside, `summarise()` and `summarize()` are equivalent and we can use either.

```
australian_politicians |>
  summarize(youngest = min(age, na.rm = TRUE),
            oldest = max(age, na.rm = TRUE),
            average = mean(age, na.rm = TRUE))
#> # A tibble: 1 x 3
#>   youngest  oldest  average
#>       <dbl>    <dbl>    <dbl>
#> 1        28     193     101.
```

By default, `summarise()` will provide one row of output for a whole dataset. For instance, in the earlier example we found the youngest, oldest, and average across all politicians. However, we can create more groups in our dataset using `group_by()`. And we can then apply another function within the context of those groups. We could use many functions on the basis of groups, but the `summarise()` function is particularly powerful in conjunction with `group_by()`. For instance, we could group by gender, and then get age-based summary statistics.

```
australian_politicians |>
  group_by(gender) |>
  summarise(youngest = min(age, na.rm = TRUE),
             oldest = max(age, na.rm = TRUE),
             average = mean(age, na.rm = TRUE))
#> # A tibble: 2 x 4
#>   gender youngest oldest average
#>   <chr>     <dbl>   <dbl>    <dbl>
#> 1 female      32     140     66.0
#> 2 male        28     193    106.
```

Similarly, we could look at youngest, oldest, and mean age at death by gender.

```
australian_politicians |>
  mutate(days_lived = deathDate - birthDate) |>
  filter(!is.na(days_lived)) |>
  group_by(gender) |>
  summarise(
    min_days = min(days_lived),
    mean_days = mean(days_lived) |> round(),
    max_days = max(days_lived)
  )
#> # A tibble: 2 x 4
#>   gender min_days  mean_days max_days
#>   <chr>    <dttm>    <dttm>    <dttm>
#> 1 female 14856 days 28857 days 35560 days
#> 2 male   12380 days 27376 days 36416 days
```

And so we learn that female members of parliament on average lived slightly longer than male members of parliament.

We can use `group_by()` on the basis of more than one group. For instance, we could look at the average number of days lived by gender and by house.

```
australian_politicians |>
  mutate(days_lived = deathDate - birthDate) |>
  filter(!is.na(days_lived)) |>
  group_by(gender, member) |>
  summarise(
    min_days = min(days_lived),
    mean_days = mean(days_lived) |> round(),
    max_days = max(days_lived)
  )
#> # A tibble: 4 x 5
```

```
#> # Groups: gender [2]
#>   gender member min_days  mean_days  max_days
#>   <chr>    <dbl> <drttn>    <drttn>    <drttn>
#> 1 female      0 21746 days 29517 days 35560 days
#> 2 female      1 14856 days 27538 days 33442 days
#> 3 male        0 13619 days 27133 days 36416 days
#> 4 male        1 12380 days 27496 days 36328 days
```

We can use `count()` to create counts by groups. For instance, the number of politicians by gender.

```
australian_politicians |>
  group_by(gender) |>
  count()
#> # A tibble: 2 x 2
#>   gender     n
#>   <chr> <int>
#> 1 female    240
#> 2 male      1543
```

In addition to the `count()`, we could make a proportion.

```
australian_politicians |>
  group_by(gender) |>
  count() |>
  ungroup() |>
  mutate(proportion = n/(sum(n)))
#> # A tibble: 2 x 3
#>   gender     n proportion
#>   <chr> <int>      <dbl>
#> 1 female    240      0.135
#> 2 male      1543     0.865
```

Using `count()` is essentially the same as using `group_by()` and then `summarise()`, and we get the same result in that way.

```
australian_politicians |>
  group_by(gender) |>
  summarise(n = n())
#> # A tibble: 2 x 2
#>   gender     n
#>   <chr> <int>
```

```
#> 1 female    240
#> 2 male     1543
```

And there is a similarly helpful function for `mutate()`, which is `add_count()`. The difference is that the number will be added in a column.

```
australian_politicians |>
  group_by(gender) |>
  add_count() |>
  select(uniqueID, gender, n)
#> # A tibble: 1,783 x 3
#> # Groups:   gender [2]
#>   uniqueID   gender     n
#>   <chr>      <chr>    <int>
#> 1 Abbott1859 male    1543
#> 2 Abbott1869 male    1543
#> 3 Abbott1877 male    1543
#> 4 Abbott1886 male    1543
#> 5 Abbott1891 male    1543
#> 6 Abbott1957 male    1543
#> 7 Abel1939 male    1543
#> 8 Abetz1958 male    1543
#> 9 Adams1943 female   240
#> 10 Adams1951 male    1543
#> # ... with 1,773 more rows
```

3.6 Base

While the `tidyverse` was established relatively recently to help with data science, R existed long before this. There is a host of functionality that is built into R especially around the core needs of programming and statisticians.

In particular, we will cover:

1. `class()`
2. data simulation
3. `function()`, `for()`, and `apply()`

There is no need to install any additional packages, as this functionality comes with R.

3.6.1 `class()`

In everyday usage ‘a, b, c, ...’ are letters and ‘1, 2, 3,...’ are numbers. And we use letters and numbers differently, for instance we do not add letters. Similarly, R needs to have some way of distinguishing different classes of content. And to define the properties that each class has, ‘how it behaves, and how it relates to other types of objects’ (Wickham, 2019a).

Classes have a hierarchy. For instance, we are ‘human’, which is itself ‘animal’. All ‘humans’ are ‘animals’, but not all ‘animals’ are ‘humans’. Similarly, all integers are numbers, but not all numbers are integers. We can find out the class of an object in R with `class()`.

```
a_number <- 8
class(a_number)
#> [1] "numeric"

a_letter <- "a"
class(a_letter)
#> [1] "character"
```

The classes that we cover here are ‘numeric’, ‘character’, ‘factor’, ‘date’, and ‘data.frame’.

The first thing to know is that, in the same way that a frog can become a prince, we can sometimes change the class of an object in R. For instance, we could start with a ‘numeric’, change it to a ‘character’ with `as.character()`, and then a ‘factor’ with `as.factor()`. But if we tried to make it into a date with `as.Date()` we would get an error because no all numbers have the properties that are needed to be a date.

```
a_number <- 8
a_number
#> [1] 8
class(a_number)
#> [1] "numeric"

a_number <- as.character(a_number)
a_number
#> [1] "8"
class(a_number)
#> [1] "character"

a_number <- as.factor(a_number)
a_number
```

```
#> [1] 8
#> Levels: 8
class(a_number)
#> [1] "factor"
```

Compared with ‘numeric’ and ‘character’ classes, the ‘factor’ class might be less familiar. A ‘factor’ is used for categorical data that can only take certain values (Wickham, 2019a). For instance, typical usage of a factor variable would be a binary, such as ‘day’ or ‘night’. It is also often used for age-groups, such as ‘18-29’, ‘30-44’, ‘45-60’, ‘60+’ (as opposed to age, which would often be a ‘numeric’); and sometimes for level of education: ‘less than high school’, ‘high school’, ‘college’, ‘undergraduate degree’, ‘postgraduate degree’. We can find the allowed levels for a ‘factor’ using `levels()`.

```
age_groups <- factor(
  c('18-29', '30-44', '45-60', '60+')
)
age_groups
#> [1] 18-29 30-44 45-60 60+
#> Levels: 18-29 30-44 45-60 60+
class(age_groups)
#> [1] "factor"
levels(age_groups)
#> [1] "18-29" "30-44" "45-60" "60+"
```

Dates are an especially tricky class and quickly become complicated. Nonetheless, at a foundational level, we can use `as.Date()` to convert a character that looks like a date into an actual date. This enables us to, say, perform addition and subtraction, when we would not be able to do that with a character.

```
looks_like_a_date_but_is_not <- "2022-01-01"
looks_like_a_date_but_is_not
#> [1] "2022-01-01"
class(looks_like_a_date_but_is_not)
#> [1] "character"
is_a_date <- as.Date(looks_like_a_date_but_is_not)
is_a_date
#> [1] "2022-01-01"
class(is_a_date)
#> [1] "Date"
is_a_date + 3
#> [1] "2022-01-04"
```

The final class that we discuss here is ‘`data.frame`’. This looks like a spreadsheet

and is commonly used to store the data that we will analyze. Formally, ‘a data frame is a list of equal-length vectors’ (Wickham, 2019a). It will have column and row names which we can see using `colnames()` and `rownames()`, although often the names of the rows are just numbers.

To illustrate this, we use the ‘ResumeNames’ dataset from `AER` (Kleiber and Zeileis, 2008). This package can be installed in the same way as any other package from CRAN. This dataset comprises cross-sectional data about resume content, especially the name used on the resume, and associated information about whether the candidate received a call-back for 4,870 fictitious resumes. The dataset was created by Bertrand and Mullainathan (2004) who sent fictitious resumes in response to job advertisements in Boston and Chicago that differed in whether the resume was assigned a ‘very African American sounding name or a very White sounding name’. They found considerable discrimination whereby ‘White names receive 50 percent more callbacks for interviews’.

```
install.packages("AER")

library(AER)
data("ResumeNames", package = "AER")

ResumeNames |>
  head()
#>      name gender ethnicity quality call     city jobs
#> 1 Allison female    cauc     low   no chicago    2
#> 2 Kristen female    cauc    high   no chicago    3
#> 3 Lakisha female    afam     low   no chicago    1
#> 4 Latonya female    afam    high   no chicago    4
#> 5 Carrie female    cauc    high   no chicago    3
#> 6 Jay male        cauc     low   no chicago    2
#>   experience honors volunteer military holes school email
#> 1          6 no       no       no yes   no   no
#> 2          6 no       yes      yes no yes yes
#> 3          6 no       no       no no yes no
#> 4          6 no       yes      no yes no yes
#> 5         22 no       no       no no yes yes
#> 6          6 yes      no       no no no no
#>   computer special college minimum equal      wanted
#> 1 yes      no       yes      5 yes supervisor
#> 2 yes      no       no       5 yes supervisor
#> 3 yes      no       yes      5 yes supervisor
#> 4 yes      yes      no       5 yes supervisor
#> 5 yes      no       no       some yes secretary
```

```
#> 6      no     yes     yes     none    yes     other
#> requirements reqexp reqcomm reqeduc reqcomp reqorg
#> 1      yes    yes     no      no      yes    no
#> 2      yes    yes     no      no      yes    no
#> 3      yes    yes     no      no      yes    no
#> 4      yes    yes     no      no      yes    no
#> 5      yes    yes     no      no      yes    yes
#> 6      no     no      no      no      no     no
#>                               industry
#> 1                           manufacturing
#> 2                           manufacturing
#> 3                           manufacturing
#> 4                           manufacturing
#> 5   health/education/social services
#> 6                           trade
class(ResumeNames)
#> [1] "data.frame"
colnames(ResumeNames)
#> [1] "name"          "gender"        "ethnicity"
#> [4] "quality"       "call"          "city"
#> [7] "jobs"          "experience"    "honors"
#> [10] "volunteer"    "military"      "holes"
#> [13] "school"        "email"         "computer"
#> [16] "special"       "college"       "minimum"
#> [19] "equal"          "wanted"        "requirements"
#> [22] "reqexp"        "reqcomm"       "reqeduc"
#> [25] "reqcomp"       "reqorg"        "industry"
```

We can examine the class of the vectors, i.e. the columns, that make-up a data frame by specifying the column name.

```
class(ResumeNames$name)
#> [1] "factor"
class(ResumeNames$jobs)
#> [1] "integer"
```

Sometimes it is helpful to be able to change the classes of many columns at once. We can do this by using `mutate()` and `across()`.

```
class(ResumeNames$name)
#> [1] "factor"
class(ResumeNames$gender)
#> [1] "factor"
```

```

class(ResumeNames$ethnicity)
#> [1] "factor"

ResumeNames |>
  mutate(across(c(name, gender, ethnicity), as.character)) |>
  head()

#>      name gender ethnicity quality call    city jobs
#> 1 Allison female    cauc     low no chicago 2
#> 2 Kristen female   cauc     high no chicago 3
#> 3 Lakisha female  afam     low no chicago 1
#> 4 Latonya female  afam     high no chicago 4
#> 5 Carrie female   cauc     high no chicago 3
#> 6 Jay male        cauc     low no chicago 2
#> experience honors volunteer military holes school email
#> 1       6 no         no       no yes   no   no
#> 2       6 no         yes      yes no   yes yes
#> 3       6 no         no       no no   yes no
#> 4       6 no         yes      no yes   no yes
#> 5      22 no         no       no no   yes yes
#> 6       6 yes        no       no no   no   no
#> computer special college minimum equal    wanted
#> 1 yes no yes 5 yes supervisor
#> 2 yes no no 5 yes supervisor
#> 3 yes no yes 5 yes supervisor
#> 4 yes yes no 5 yes supervisor
#> 5 yes no no some yes secretary
#> 6 no yes yes none yes other
#> requirements reqexp reqcomm reqeduc reqcomp reqorg
#> 1 yes yes no no yes no
#> 2 yes yes no no yes no
#> 3 yes yes no no yes no
#> 4 yes yes no no yes no
#> 5 yes yes no no yes yes
#> 6 no no no no no no
#> industry
#> 1 manufacturing
#> 2 manufacturing
#> 3 manufacturing
#> 4 manufacturing
#> 5 health/education/social services
#> 6 trade

class(ResumeNames$name)
#> [1] "factor"

```

```
class(ResumeNames$gender)
#> [1] "factor"
class(ResumeNames$ethnicity)
#> [1] "factor"
```

There are many ways for code to not run but having an issue with the class is always among the first things to check. Common issues are variables that we think should be ‘character’ or ‘numeric’, actually being ‘factor’. And variables that we think should be ‘numeric’ actually being ‘character’.

3.6.2 Simulating data

Simulating data is a key skill for telling believable stories with data. In order to simulate data, we need to be able to randomly draw from statistical distributions and other collections. R has a variety of functions to make this easier, including: the normal distribution, `rnorm()`; the uniform distribution, `runif()`; the Poisson distribution, `rpois`; the binomial distribution, `rbinom`; and many others. To randomly sample from a collection of items, we can use `sample()`.

When dealing with randomness, the need for reproducibility makes it important, paradoxically, that the randomness is repeatable. That is to say, another person needs to be able to draw the random numbers that we draw. We do this by setting a seed for our random draws using `set.seed()`.

We could get observations from the standard normal distribution and put those into a data frame.

```
set.seed(853)

number_of_observations <- 5

simulated_data <-
  data.frame(
    person = c(1:number_of_observations),
    std_normal_observations = rnorm(n = number_of_observations,
                                    mean = 0,
                                    sd = 1)
  )

simulated_data
#>   person std_normal_observations
#> 1       1             -0.35980342
#> 2       2             -0.04064753
#> 3       3             -1.78216227
```

```
#> 4      4      -1.12242282
#> 5      5      -1.00278400
```

We could then add draws from the uniform, Poisson, and binomial distributions, using `cbind()` to bring the columns of the original dataset and the new one together.

```
simulated_data <-
  data.frame(
    uniform_observations =
      runif(n = number_of_observations, min = 0, max = 10),
    poisson_observations =
      rpois(n = number_of_observations, lambda = 100),
    binomial_observations =
      rbinom(n = number_of_observations, size = 2, prob = 0.5)
  ) |>
  cbind(simulated_data)

simulated_data
#>   uniform_observations poisson_observations
#> 1      9.6219155          81
#> 2      7.2269016          91
#> 3      0.8252921          84
#> 4      1.0379810          100
#> 5      3.0942004          97
#>   binomial_observations person std_normal_observations
#> 1                  2     1      -0.35980342
#> 2                  1     2      -0.04064753
#> 3                  1     3      -1.78216227
#> 4                  1     4      -1.12242282
#> 5                  1     5      -1.00278400
```

Finally, we will add a favorite color to each observation with `sample()`.

```
simulated_data <-
  data.frame(
    favorite_color = sample(x = c("blue", "white"),
                           size = number_of_observations,
                           replace = TRUE)
  ) |>
  cbind(simulated_data)

simulated_data
```

```
#>   favorite_color uniform_observations poisson_observations
#> 1       blue      9.6219155          81
#> 2       blue      7.2269016          91
#> 3       blue      0.8252921          84
#> 4       white     1.0379810         100
#> 5       blue      3.0942004          97
#>   binomial_observations person std_normal_observations
#> 1             2     1      -0.35980342
#> 2             1     2      -0.04064753
#> 3             1     3      -1.78216227
#> 4             1     4      -1.12242282
#> 5             1     5      -1.00278400
```

We set the option ‘replace’ to ‘TRUE’ because we are only choosing between two items, but each time we choose we want the possibility that either are chosen. Depending on the simulation we may need to think about whether ‘replace’ should be ‘TRUE’ or ‘FALSE’. Another useful optional argument in `sample()` is to adjust the probability with which each item is drawn. The default is that all options are equally likely, but we could specify particular probabilities if we wanted to with ‘prob’. As always with functions, we can find more in the help file, for instance `?sample`.

3.6.3 `function()`, `for()`, and `apply()`

R ‘is a functional programming language’ (Wickham, 2019a). This means that we foundationally write, use, and compose functions, which are collections of code that accomplish something specific.

There are a lot of functions in R that other people have written, and we can use. Almost any common statistical or data science task that we might need to accomplish likely already has a function that has been written by someone else and made available to us, either as part of the base R installation or a package. But we will need to write our own functions from time to time, especially for more-specific tasks. We define a function using `function()`, and then assign a name. We will likely need to include some inputs and outputs for the function. Inputs are specified between round brackets. The specific task that the function is to accomplish goes between braces.

```
print_names <- function(some_names) {
  print(some_names)
}

print_names(c("rohan", "monica"))
#> [1] "rohan"  "monica"
```



```

#> 2          1     2      -0.04064753
#> 3          1     3      -1.78216227
#> 4          1     4      -1.12242282
#> 5          1     5      -1.00278400
apply(X = simulated_data, MARGIN = 2, FUN = unique)
#> $favorite_color
#> [1] "blue"    "white"
#>
#> $uniform_observations
#> [1] "9.6219155" "7.2269016" "0.8252921" "1.0379810"
#> [5] "3.0942004"
#>
#> $poisson_observations
#> [1] "81" "91" "84" "100" "97"
#>
#> $binomial_observations
#> [1] "2" "1"
#>
#> $person
#> [1] "1" "2" "3" "4" "5"
#>
#> $std_normal_observations
#> [1] "-0.35980342" "-0.04064753" "-1.78216227" "-1.12242282"
#> [5] "-1.00278400"

```

3.7 Making graphs with `ggplot2`

If the key package in the `tidyverse` in terms of manipulating data is `dplyr` (Wickham et al., 2020a), then the key package in the `tidyverse` in terms of creating graphs is `ggplot2` (Wickham, 2016). It is part of the `tidyverse` collection of packages and so does not need to be explicitly installed or loaded if the `tidyverse` has been loaded.

More formally, `ggplot2` works by defining layers which build to form a graph, based around the ‘grammar of graphics’ (hence, the ‘gg’). Instead of the pipe operator (`|>`) `ggplot` uses the add operator `+`.

There are three key aspects that need to be specified to build a graph with `ggplot2`:

1. data;
2. aesthetics / mapping; and

3. type.

To get started we will obtain some GDP data for OECD countries ([OECD, 2022](#)).

```
library(tidyverse)

oecd_gdp <-
  read_csv("https://stats.oecd.org/sdmx-json/data/DP_LIVE/.QGDP.../OECD?contentType=csv&detail=cod")

write_csv(oecd_gdp, 'inputs/data/oecd_gdp.csv')

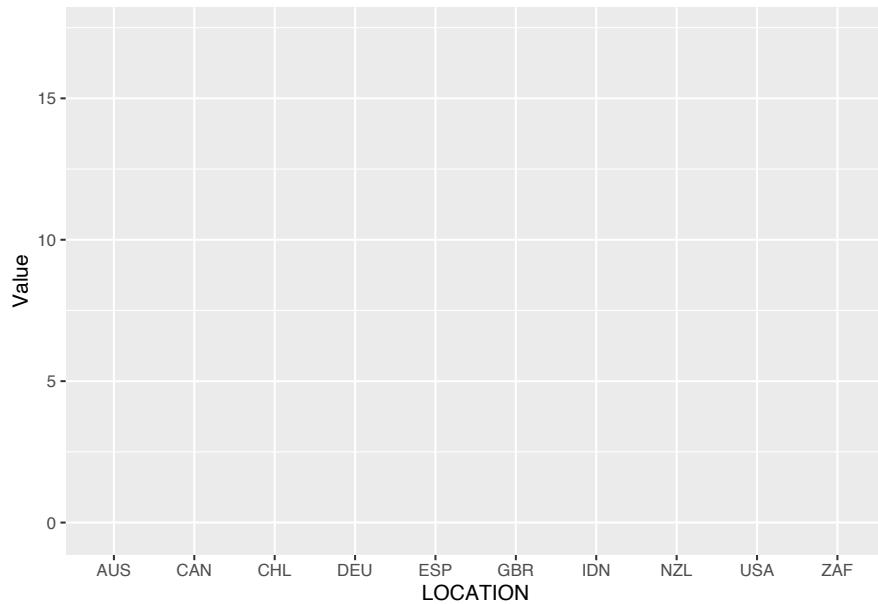
#> # A tibble: 6 x 8
#>   LOCATION INDICATOR SUBJECT MEASURE FREQUENCY TIME  Value
#>   <chr>     <chr>    <chr>   <chr>    <chr>    <chr> <dbl>
#> 1 OECD       QGDP      TOT     PC_CHGPP A      1962    5.70
#> 2 OECD       QGDP      TOT     PC_CHGPP A      1963    5.20
#> 3 OECD       QGDP      TOT     PC_CHGPP A      1964    6.38
#> 4 OECD       QGDP      TOT     PC_CHGPP A      1965    5.35
#> 5 OECD       QGDP      TOT     PC_CHGPP A      1966    5.75
#> 6 OECD       QGDP      TOT     PC_CHGPP A      1967    3.96
#> # ... with 1 more variable: Flag Codes <chr>
```

We are interested, firstly, in making a bar chart of GDP change in the third quarter of 2021 for ten countries: Australia, Canada, Chile, Indonesia, Germany, Great Britain, New Zealand, South Africa, Spain, and the US.

```
oecd_gdp_most_recent <-
  oecd_gdp |>
  filter(TIME == "2021-Q3",
         SUBJECT == "TOT",
         LOCATION %in% c("AUS", "CAN", "CHL", "DEU", "GBR",
                         "IDN", "ESP", "NZL", "USA", "ZAF"),
         MEASURE == "PC_CHGPP") |>
  mutate(european = if_else(LOCATION %in% c("DEU", "GBR", "ESP"),
                           "European",
                           "Not european"),
         hemisphere = if_else(LOCATION %in% c("CAN", "DEU", "GBR", "ESP", "USA"),
                           "Northern Hemisphere",
                           "Southern Hemisphere"),
         )
```

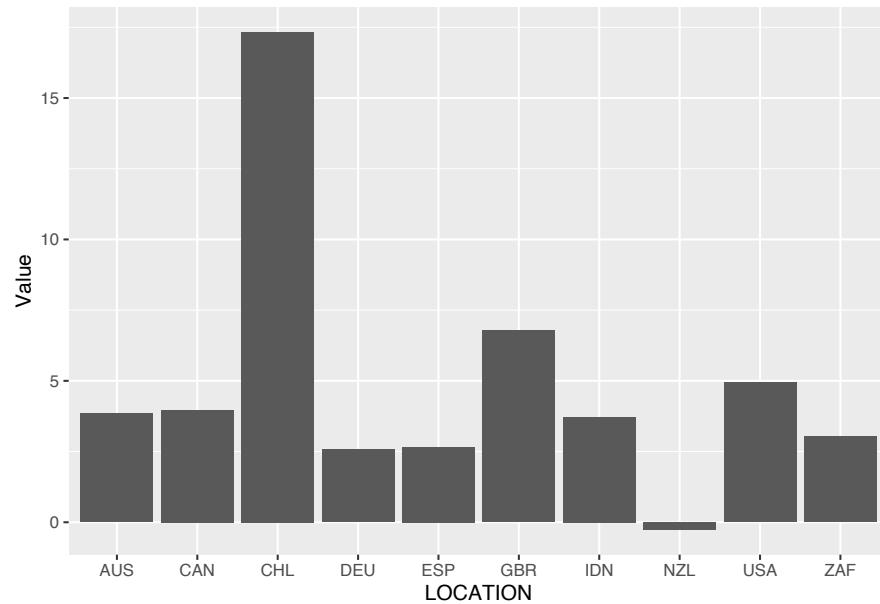
We start with `ggplot` and specify a mapping/aesthetic, which in this case means specifying the x-axis and the y-axis.

```
oecd_gdp_most_recent |>  
  ggplot(mapping = aes(x = LOCATION, y = Value))
```



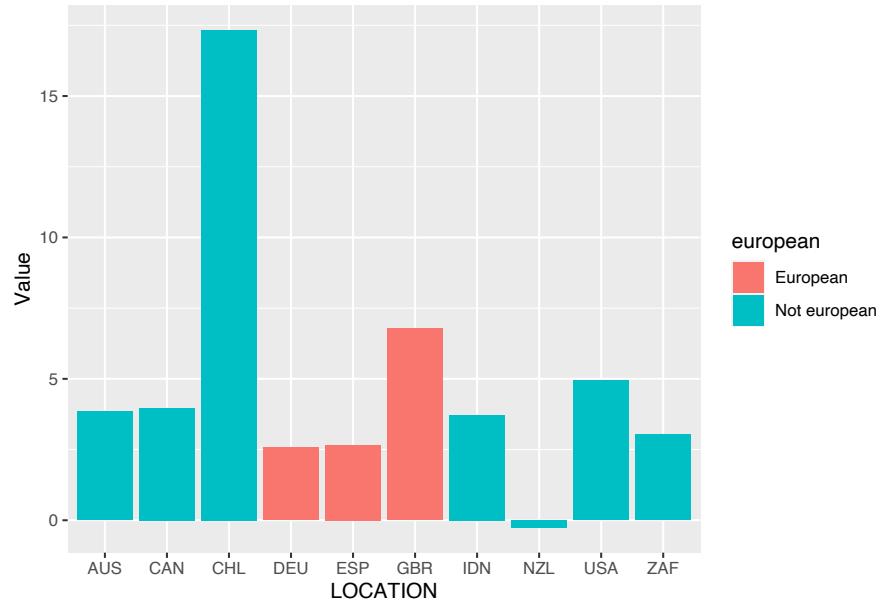
Now we need to specify the type of graph that we are interested in. In this case we want a bar chart and we do this by adding `geom_bar()`.

```
oecd_gdp_most_recent |>  
  ggplot(mapping = aes(x = LOCATION, y = Value)) +  
    geom_bar(stat="identity")
```



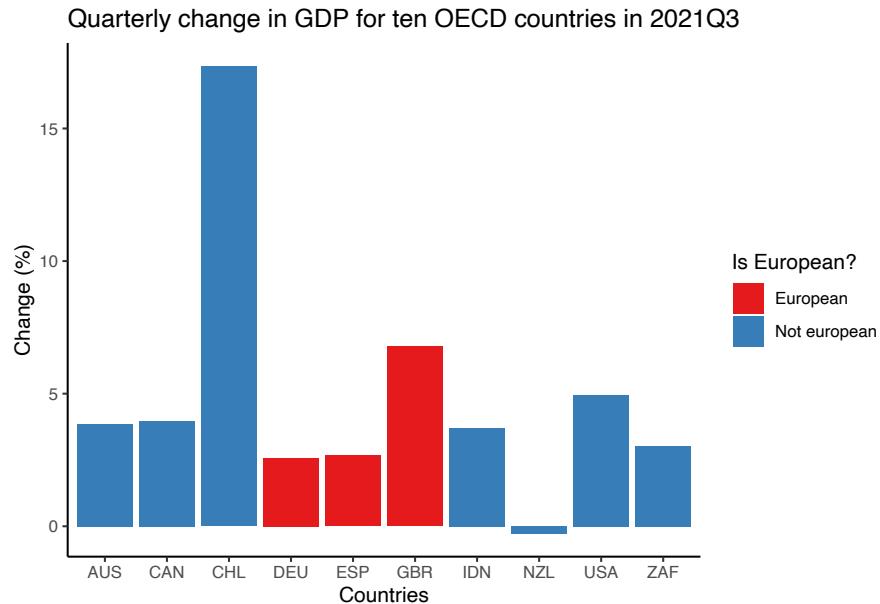
We can color the bars by whether the country is European by adding another aesthetic, ‘fill’.

```
oecd_gdp_most_recent |>
  ggplot(mapping = aes(x = LOCATION, y = Value, fill = european)) +
  geom_bar(stat="identity")
```



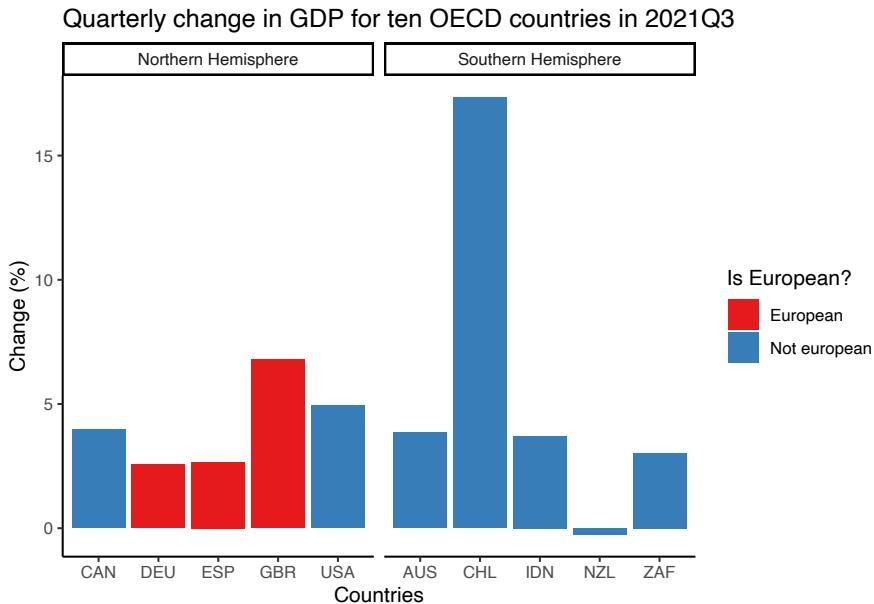
Finally, we could make it look nicer by: adding labels, `labs()`; changing the color, `scale_fill_brewer()`; and the background, `theme_classic()`.

```
oecd_gdp_most_recent |>
  ggplot(mapping = aes(x = LOCATION, y = Value, fill = european)) +
  geom_bar(stat="identity") +
  labs(title = "Quarterly change in GDP for ten OECD countries in 2021Q3",
       x = "Countries",
       y = "Change (%)",
       fill = "Is European?") +
  theme_classic() +
  scale_fill_brewer(palette = "Set1")
```



Facets enable us to that we create subplots that focus on specific aspects of our data. They are invaluable because they allow us to add another variable to a graph without having to make a 3D graph. We use `facet_wrap()` to add a facet and specify the variable that we would like to facet by. In this case, we facet by hemisphere.

```
oecd_gdp_most_recent |>
  ggplot(mapping = aes(x = LOCATION, y = Value, fill = european)) +
  geom_bar(stat="identity") +
  labs(title = "Quarterly change in GDP for ten OECD countries in 2021Q3",
       x = "Countries",
       y = "Change (%)",
       fill = "Is European?") +
  theme_classic() +
  scale_fill_brewer(palette = "Set1") +
  facet_wrap(~hemisphere,
             scales = "free_x")
```



3.8 Exploring the tidyverse

We have focused on two aspects of the tidyverse: `dplyr`, and `ggplot2`. However, the tidyverse comprises a variety of different packages and functions. We will now go through four common aspects:

- Importing data and `tibble()`
- Joining and pivoting datasets
- String manipulation and `stringr`
- Factor variables and `forcats`

3.8.1 Importing data and `tibble()`

There are a variety of ways to get data into R so that we can use it. For CSV files, there is `read_csv()` from `readr` (Wickham et al., 2021), and for dta files, there is `read_dta()` from `haven` (Wickham and Miller, 2020).

CSVs are a common format and have many advantages including the fact that they typically do not modify the data. Each column is separated by a comma, and each row is a record. We can provide `read_csv()` with a URL or a local file to read. There are a variety of different options that can be passed to `read_csv()` including the ability to specify whether the dataset has column

names, the types of the columns, and how many lines to skip. If we do not specify the types of the columns then `read_csv()` will make a guess by looking at the dataset.

We use `read_dta()` to read .dta files, which are commonly produced by the statistical program Stata. This means that they are common in fields such as sociology, political science, and economics. This format separates the data from its labels and so we typically reunite these using `to_factor()` from `labelled` (Larmarange, 2021). `haven` is part of the `tidyverse`, but is not automatically loaded by default, in contrast to a package such as `ggplot2`, and so we would need to run `library(haven)`.

Typically a dataset enters R as a ‘`data.frame`’. While this can be useful, another helpful class for a dataset is ‘`tibble`’. These can be created using `tibble()` from the `tibble` package which is part of the `tidyverse`. A `tibble` is a data frame, with some particular changes that make it easier to work with, including not converting strings to factors by default, showing the class of columns, and printing nicely.

We can make a `tibble` manually, if need be, for instance, when we simulate data. But we typically import data directly as a `tibble`, for instance, when we use `read_csv()`.

```
people_as_dataframe <-
  data.frame(names = c("rohan", "monica"),
             website = c("rohanalexander.com", "monicaalexander.com"),
             fav_color = c("blue", "white"))
  )
class(people_as_dataframe)
#> [1] "data.frame"
people_as_dataframe
#>   names           website fav_color
#> 1 rohan rohanalexander.com     blue
#> 2 monica monicaalexander.com    white

people_as_tibble <-
  tibble(names = c("rohan", "monica"),
         website = c("rohanalexander.com", "monicaalexander.com"),
         fav_color = c("blue", "white"))
  )
people_as_tibble
#> # A tibble: 2 x 3
#>   names   website      fav_color
#>   <chr>   <chr>        <chr>
#> 1 rohan rohanalexander.com "blue"
#> 2 monica monicaalexander.com "white"
```

```
class(people_as_tibble)
#> [1] "tbl_df"     "tbl"        "data.frame"
```

3.8.2 Dataset manipulation with joins and pivots

There are two dataset manipulations that are often needed: joins and pivots.

We often have a situation where we have two, or more, datasets and we are interested in combining them. We can join datasets together in a variety of ways. A common way is to use `left_join()` from `dplyr` (Wickham et al., 2020a). This is most useful where there is one main dataset that we are using and there is another dataset with some useful variables that we want to add to that. The critical aspect is that we have column/s that we can use to link the two datasets. Here we will create two tibbles and then join them on the basis of names.

```
main_dataset <-
  tibble(
    names = c('rohan', 'monica', 'edward', 'hugo'),
    status = c('adult', 'adult', 'child', 'infant')
  )
main_dataset
#> # A tibble: 4 x 2
#>   names  status
#>   <chr> <chr>
#> 1 rohan adult
#> 2 monica adult
#> 3 edward child
#> 4 hugo   infant

supplementary_dataset <-
  tibble(
    names = c('rohan', 'monica', 'edward', 'hugo'),
    favorite_food = c('pasta', 'salmon', 'pizza', 'milk')
  )
supplementary_dataset
#> # A tibble: 4 x 2
#>   names  favorite_food
#>   <chr> <chr>
#> 1 rohan pasta
#> 2 monica salmon
#> 3 edward pizza
#> 4 hugo   milk
```

```
main_dataset <-
  main_dataset |>
  left_join(supplementary_dataset, by = "names")

main_dataset
#> # A tibble: 4 x 3
#>   names  status favorite_food
#>   <chr>  <chr>   <chr>
#> 1 rohan  adult   pasta
#> 2 monica adult   salmon
#> 3 edward child   pizza
#> 4 hugo   infant  milk
```

There are a variety of other options to join datasets, including `inner_join()`, `right_join()`, and `full_join()`.

Another common dataset manipulation task is pivoting them. Datasets tend to be either long or wide. Generally, in the `tidyverse`, and certainly for `ggplot2`, we need long data. To go from one to the other we use `pivot_longer()` and `pivot_wider()` from `tidyverse` ([Wickham, 2021b](#)).

We will create some wide data on whether ‘mark’ or ‘lauren’ won a running race in each of three years.

```
pivot_example_data <-
  tibble(year = c(2019, 2020, 2021),
        mark = c("first", "second", "first"),
        lauren = c("second", "first", "second"))

pivot_example_data
#> # A tibble: 3 x 3
#>   year  mark  lauren
#>   <dbl> <chr> <chr>
#> 1 2019 first second
#> 2 2020 second first
#> 3 2021 first second
```

This dataset is in wide format at the moment. To get it into long format, we need a column that specifies the person, and another that specifies the result. We use `pivot_longer()` to achieve this.

```
data_pivoted_longer <-
  pivot_example_data |>
  tidyverse::pivot_longer(cols = c("mark", "lauren"),
```

```

    names_to = "person",
    values_to = "position")

head(data_pivoted_longer)
#> # A tibble: 6 x 3
#>   year person position
#>   <dbl> <chr>  <chr>
#> 1 2019 mark   first
#> 2 2019 lauren second
#> 3 2020 mark   second
#> 4 2020 lauren first
#> 5 2021 mark   first
#> 6 2021 lauren second

```

Occasionally, we need to go from long data to wide data. We use `pivot_wider()` to do this.

```

data_pivoted_wider <-
  data_pivoted_longer |>
  tidyverse::pivot_wider(id_cols = c("year", "person"),
                        names_from = "person",
                        values_from = "position")

head(data_pivoted_wider)
#> # A tibble: 3 x 3
#>   year mark  lauren
#>   <dbl> <chr>  <chr>
#> 1 2019 first second
#> 2 2020 second first
#> 3 2021 first  second

```

3.8.3 String manipulation and `stringr`

In R we often create a string with double quotes, although using single quotes works too. For instance `c("a", "b")` consists of two strings ‘a’ and ‘b’, that are contained in a character vector. There are a variety of ways to manipulate strings in R and we focus on `stringr` (Wickham, 2019e). This is automatically loaded when we load the `tidyverse`.

If we want to look for whether a string contains certain content, then we can use `str_detect()`. And if we want to remove or change some particular content then we can use `str_remove()` or `str_replace()`.

```
dataset_of_strings <-
  tibble(
    names = c("rohan alexander",
              "monica alexander",
              "edward alexander",
              "hugo alexander")
  )

dataset_of_strings |>
  mutate(is_rohan = str_detect(names, "rohan"),
         make_howlett = str_replace(names, "alexander", "howlett"),
         remove_rohan = str_remove(names, "rohan"))
  )

#> # A tibble: 4 x 4
#>   names           is_rohan make_howlett  remove_rohan
#>   <chr>          <lgl>     <chr>        <chr>
#> 1 rohan alexander  TRUE      rohan howlett " alexander"
#> 2 monica alexander FALSE     monica howlett "monica alexande~
#> 3 edward alexander FALSE     edward howlett "edward alexande~
#> 4 hugo alexander  FALSE     hugo howlett "hugo alexander"
```

There are a variety of other functions that are often especially useful in data cleaning. For instance, we can use `str_length()` to find out how long a string is, and `str_c()` to bring strings together.

```
dataset_of_strings |>
  mutate(length_is = str_length(string = names),
         name_and_length = str_c(names, length_is, sep = " - "))
  )

#> # A tibble: 4 x 3
#>   names           length_is name_and_length
#>   <chr>          <int>     <chr>
#> 1 rohan alexander      15 rohan alexander - 15
#> 2 monica alexander      16 monica alexander - 16
#> 3 edward alexander      16 edward alexander - 16
#> 4 hugo alexander       14 hugo alexander - 14
```

Finally, `separate()` from `tidyverse`, although not part of `stringr`, is indispensable for string manipulation. It turns one character column into many.

```
dataset_of_strings |>
  separate(col = names,
           into = c("first", "last"),
```

```

      sep = " ",
      remove = FALSE)
#> # A tibble: 4 x 3
#>   names          first  last
#>   <chr>         <chr>  <chr>
#> 1 rohan alexander rohan alexander
#> 2 monica alexander monica alexander
#> 3 edward alexander edward alexander
#> 4 hugo alexander   hugo   alexander

```

3.8.4 Factor variables and `forcats`

A factor is a collection of strings that are categories. Sometimes there will be an inherent ordering. For instance, the days of the week have an order – Monday, Tuesday, Wednesday, ... – which is not alphabetical. But there is no requirement for that to be the case, for instance gender: female, male, and other; or pregnancy status: pregnant or not pregnant. Factors feature prominently in base R. They can be useful because they ensure that only appropriate strings are allowed. For instance, if ‘days_of_the_week’ was a factor variable then ‘January’ would not be allowed. But they can add a great deal of complication, and so they have a less prominent role in `tidyverse`. Nonetheless taking advantage of factors is useful in certain circumstances. For instance, when plotting the days of the week we probably want them in the usual ordering than in the alphabetical ordering that would result if we had them as a character variable. While factors are built into base R, one `tidyverse` package that is especially useful when using factors is `forcats` (Wickham, 2020a).

Sometimes we have a character vector, and we will want it ordered in a particular way. The default is that a character vector is ordered alphabetically, but we may not want that. For instance, the days of the week would look strange on a graph if they were alphabetically ordered: Friday, Monday, Saturday, Sunday, Thursday, Tuesday, and Wednesday!

The way to change the ordering is to change the variable from a character to a factor. We can use `fct_relevel()` from `forcats` (Wickham, 2020a) to specify an ordering.

```

set.seed(853)

days_data <-
tibble(
  days =
  c(

```

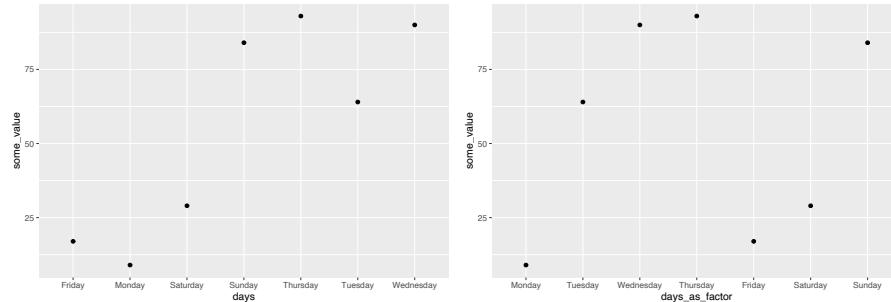
```
"Monday",
"Tuesday",
"Wednesday",
"Thursday",
"Friday",
"Saturday",
"Sunday"
),
some_value = c(sample.int(100, 7))
)

days_data <-
days_data |>
mutate(
  days_as_factor = factor(days),
  days_as_factor = fct_relevel(
    days,
    "Monday",
    "Tuesday",
    "Wednesday",
    "Thursday",
    "Friday",
    "Saturday",
    "Sunday"
  )
)
```

And we can compare the results by graphing first with the original character vector on the x-axis, and then another graph with the factor vector on the x-axis.

```
days_data |>
  ggplot(aes(x = days, y = some_value)) +
  geom_point()

days_data |>
  ggplot(aes(x = days_as_factor, y = some_value)) +
  geom_point()
```



3.9 Exercises and tutorial

3.9.1 Exercises

1. What is R?
 - a. A open-source statistical programming language
 - b. A programming language created by Guido van Rossum
 - c. A closed source statistical programming language
 - d. An integrated development environment (IDE)
2. What are three advantages of R? What are three disadvantages?
3. What is R Studio?
 - a. An integrated development environment (IDE).
 - b. A closed source paid program.
 - c. A programming language created by Guido van Rossum
 - d. A statistical programming language.
4. What is the class of the output of '2 + 2' (pick one)?
 - a. character
 - b. factor
 - c. numeric
 - d. date
5. Say we had run: `my_name <- 'Rohan'`. What would be the result of running `print(my_name)` (pick one)?
 - a. 'Edward'
 - b. 'Monica'
 - c. 'Hugo'
 - d. 'Rohan'
6. Say we had a dataset with two columns: 'name', and 'age'. Which verb should we use to pick just 'name' (pick one)?
 - a. `tidyverse::select()`
 - b. `tidyverse::mutate()`
 - c. `tidyverse::filter()`
 - d. `tidyverse::rename()`

7. Say we had loaded `AustralianPoliticians` and `tidyverse` and then run the following code: `australian_politicians <- AustralianPoliticians::get_auspol('all')`. How could we select all of the columns that end with 'Name' (pick one)?
 - a. `australian_politicians |> select(contains("Name"))`
 - b. `australian_politicians |> select(starts_with("Name"))`
 - c. `australian_politicians |> select(matches("Name"))`
 - d. `australian_politicians |> select(ends_with("Name"))`
8. Under what circumstances, in terms of the names of the columns, would the use of 'contains()' potentially give different answers to using 'ends_with()' in the above question?
9. Which of the following are not `tidyverse` verbs (pick one)?
 - a. `select()`
 - b. `filter()`
 - c. `arrange()`
 - d. `mutate()`
 - e. `visualize()`
10. Which function would make a new column (pick one)?
 - a. `select()`
 - b. `filter()`
 - c. `arrange()`
 - d. `mutate()`
 - e. `visualize()`
11. Which function would focus on particular rows (pick one)?
 - a. `select()`
 - b. `filter()`
 - c. `arrange()`
 - d. `mutate()`
 - e. `summarise()`
12. Which combination of two functions could provide a mean of a dataset, by sex (pick two)?
 - a. `summarise()`
 - b. `filter()`
 - c. `arrange()`
 - d. `mutate()`
 - e. `group_by()`
13. Assume a variable called 'age' is an integer. Which line of code would create a column that is its exponential (pick one)?
 - a. `mutate(exp_age = exponential(age))`
 - b. `mutate(exp_age = exponent(age))`
 - c. `mutate(exp_age = exp(age))`
 - d. `mutate(exp_age = expon(age))`
14. Assume a column called 'age'. Which line of code could create a column that contains the value from five rows above?
 - a. `mutate(five_before = lag(age))`

- b. `mutate(five_before = lead(age))`
 - c. `mutate(five_before = lag(age, n = 5))`
 - d. `mutate(five_before = lead(age, n = 5))`
15. What would be the output of `class('edward')` (pick one)?
- a. ‘numeric’
 - b. ‘character’
 - c. ‘data.frame’
 - d. ‘vector’
16. Which function would enable us to draw once from three options ‘blue, white, red’, with 10 per cent probability on ‘blue’ and ‘white’, and the remainder on ‘red’?
- a. `sample(c('blue', 'white', 'red'), prob = c(0.1, 0.1, 0.8))`
 - b. `sample(c('blue', 'white', 'red'), size = 1)`
 - c. `sample(c('blue', 'white', 'red'), size = 1, prob = c(0.8, 0.1, 0.1))`
 - d. `sample(c('blue', 'white', 'red'), size = 1, prob = c(0.1, 0.1, 0.8))`
17. Which code simulates 10,000 draws from a normal distribution with a mean of 27 and a standard deviation of 3 (pick one)?
- a. `rnorm(10000, mean = 27, sd = 3)`
 - b. `rnorm(27, mean = 10000, sd = 3)`
 - c. `rnorm(3, mean = 10000, sd = 27)`
 - d. `rnorm(27, mean = 3, sd = 1000)`
18. What are the three key aspects of the grammar of graphics (select all)?
- a. data
 - b. aesthetics
 - c. type
 - d. `geom_histogram()`

3.9.2 Tutorial

I think we should be suspicious when we find ourselves attracted to data—very, very thin and weak data—that seem to justify beliefs that have held great currency in lots of societies throughout history, in a way that is conducive to the oppression of large segments of the population

Amia Srinivasan, 22 September 2021

Reflect on the quote from Amia Srinivasan, Chichele Professor of Social and Political Theory, All Souls College, Oxford, and [D'Ignazio and Klein \(2020\)](#), especially Chapter 6, and in two pages discuss a dataset that you are familiar with.



4

Reproducible workflows

Required material

- Read *What has happened down here is the winds have changed*, (Gelman, 2016)
- Read *Good enough practices in scientific computing*, (Wilson et al., 2017)
- Watch *Overcoming barriers to sharing code*, (Alexander, 2021)
- Watch *Make a reprex... Please*, (Gelfand, 2021)
- Read *The tidyverse style guide*, ‘Part: Analyses’, (Wickham, 2021c)

Key concepts and skills

- Reproducibility is a requirement, and this implies sharing data, code, and environment.
- Reproducibility is enhanced by using R Markdown, R Projects, and Git and GitHub.
- R Markdown involves marking text as certain types and then building the document.
- R Projects enable a file structure that is not dependent on a specific directory set-up.
- Git and GitHub make it easier to share code and data.
 - Get the latest changes: `git pull`.
 - Add your updates: `git add -A`.
 - Check on everything: `git status`.
 - Commit your changes: `git commit -m "Short description of changes"`.
 - Push your changes to GitHub: `git push`.
- Restart R often ('Session' -> 'Restart R and Clear Output').
- Debugging is a skill, that improves with practice.
- One key debugging skill is being able to make a reproducible example that reproduces the issue for others.
- Appropriate code structure and comments are a critical aspect of reproducibility because they help others understand.

4.1 Introduction

Suppose you have cancer and you have to choose between a black box AI surgeon that cannot explain how it works but has a 90% cure rate and a human surgeon with an 80% cure rate. Do you want the AI surgeon to be illegal?

Geoffrey Hinton, 20 February 2020.

The number one thing to keep in mind about machine learning is that performance is evaluated on samples from one dataset, but the model is used in production on samples that may not necessarily follow the same characteristics... The finance industry has a saying for this: “past performance is no guarantee of future results”. Your model scoring X on your test dataset doesn’t mean it will perform at level X on the next N situations it encounters in the real world. The future may not be like the past.

So when asking the question, “would you rather use a model that was evaluated as 90% accurate, or a human that was evaluated as 80% accurate”, the answer depends on whether your data is typical per the evaluation process. Humans are adaptable, models are not. If significant uncertainty is involved, go with the human. They may have inferior pattern recognition capabilities (versus models trained on enormous amounts of data), but they understand what they do, they can reason about it, and they can improvise when faced with novelty

If every possible situation is known and you want to prioritize scalability and cost-reduction, go with the model. Models exist to encode and operationalize human cognition in well-understood situations. (“well understood” meaning either that it can be explicitly described by a programmer, or that you can amass a dataset that densely samples the distribution of possible situations – which must be static)

François Chollet, 20 February 2020.

If science is about systematically building and organizing knowledge in terms of testable explanations and predictions, then data science takes this and focuses on data. This means that building, organizing, and sharing knowledge is the critical aspect. Creating knowledge, once, in a way that only you can do it, does not meet this standard. Hence, the need for reproducible workflows for data science.

Alexander (2019c) says ‘research is reproducible if it can be reproduced exactly, given all the materials used in the study... [hence] materials need to be provided!... [M]aterials usually means data, code and software.’ The minimum requirement is that another person is able to ‘reproduce the data, methods and results (including figures, tables)’. Similarly, Gelman (2016) identifies how large an issue this is in various social sciences. The problem with work that is not reproducible, is that it does not contribute to our stock of knowledge about the world. Since Gelman (2016), a great deal of work has been done in many social sciences and the situation has improved a little, but much work remains. And the situation is similar in the life sciences (Heil et al., 2021) and computer science (Pineau et al., 2021).

Some of the examples that Gelman (2016) talks about, which turned out to not reproduce are not that important in the scheme of things. But at the same time, we saw, and continue to see, similar approaches being used in areas with big impacts. For instance, many governments have created ‘nudge’ units that implement public policy (Sunstein and Reisch, 2017) and they are increasingly using algorithms that they do not make open (Chouldechova et al., 2018). Similarly, businesses are increasingly implementing machine learning methods.

At a minimum, and with few exceptions, we must release our code, datasets, and environment. Without the data, we do not know what a finding speaks to (Miyakawa, 2020). More banally, we also do not know if there are mistakes or aspects that were inadvertently overlooked (Merali, 2010) (Hillel, 2017) (Silver, 2020).

To be more specific, consider Wang and Kosinski (2018) who use deep neural networks to train a model to distinguish between gay and heterosexual men. (Murphy (2017) provides a summary of the paper and the associated issues, along with comments from the authors.) To do this, Wang and Kosinski (2018, p. 248) needed a dataset of photos of folks that were ‘adult, Caucasian, fully visible, and of a gender that matched the one reported on the user’s profile’. They verified this using Amazon Mechanical Turk, an online platform that pays workers a small amount of money to complete specific tasks. Figure 4.1, from Wang and Kosinski (2018) supplemental materials, shows the instructions provided to the Mechanical Turk workers for this task. Some of the issues with these instructions include that Obama had a white mother and

a black father but has been classified as ‘Black’; and Latino is an ethnicity, rather than a race (Mattson, 2017). The classification task may seem objective, but, perhaps unthinkingly, echoes the views of Americans with a certain class and background.

This is just one specific concern about one part of the Wang and Kosinski (2018) workflow. Broader concerns are raised by others including Gelman et al. (2018). The main issue is that statistical models are specific to the data on which they were trained. And the only reason that we can identify likely issues in the model of Wang and Kosinski (2018) is because, despite not releasing the specific dataset that they used, they were nonetheless open about their procedure. For our work to be credible, it needs to be reproducible by others.

Some of the steps that we can take to make our work more reproducible include:

1. Ensure the entire workflow is documented and this may involve addressing questions such as:
 - How was the raw dataset obtained and is access likely to be persistent and available to others?
 - What specific steps are being taken to transform the raw data in the data that were analyzed, and how can this be made available to others?
 - What analysis has been done, and how clearly can this be shared?
 - How has the final paper or report been built and to what extent can others follow that process themselves?
2. Not worrying about perfect reproducibility initially, but instead focusing on trying to improve with each successive project. For instance, each of the following requirements are increasingly more onerous and there is no need to be concerned about not being able to the last, until we can do the first:
 - Can you run your entire workflow again?
 - Can ‘another person’ run your entire workflow again?
 - Can ‘future’ you run your entire workflow again?
 - Can ‘future’ ‘another person’ run your entire workflow again?
3. Including a detailed discussion about the limitations of the dataset and the approach in the final paper or report.

The workflow that we follow is summarized in Figure 4.2.

There are various tools that we can use at the different stages that will improve the reproducibility of this workflow. This includes the use of R Markdown, R Projects, and Git and GitHub.

Identify Adult Caucasian Males

Instructions

You will see 50 sets of 4 faces. Your job is to select **complete** faces belonging to **adult Caucasians males**. Any given set can contain between 0 to 4 adult male Caucasian faces.

You can use Back and Next button to navigate through different sets. Please use the best of your intuition. We will carefully review the results to identify spammers.

We welcome your feedback! There are going to be more HITs like these!

Details

1. Some images might contain a grey space on the side. It's normal and shouldn't affect your selections.
2. Some faces might be blurry. As long as you can recognize that the image represents an adult Caucasian male, the face should be accepted.
3. Faces partially covered by hats, sunglasses and hair are considered complete as long as you can recognize an adult Caucasian male.

Examples

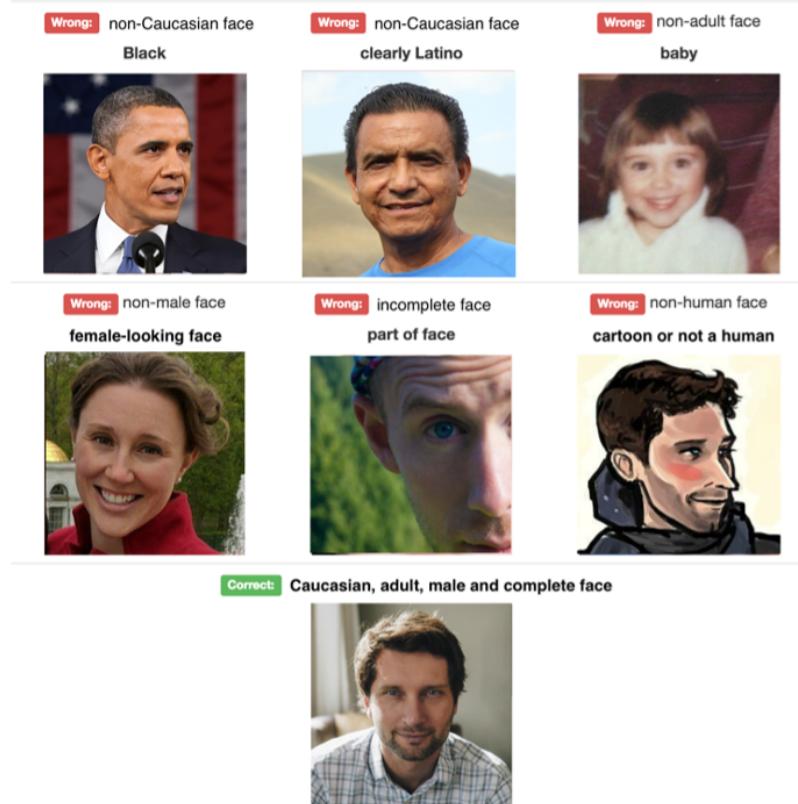


FIGURE 4.1: Instructions given to Mechanical Turk workers for removing incomplete, non-Caucasian, nonadult, and nonhuman male faces



FIGURE 4.2: Workflow for telling stories with data

4.2 R Markdown

4.2.1 Getting started

R Markdown is a mark-up language similar to HyperText Markup Language (HTML) or LaTeX, in comparison to a ‘What You See Is What You Get’ (WYSIWYG) language, such as Microsoft Word. This means that all the aspects are consistent, for instance, all top-level heading will look the same. But, it means that we must use symbols to designate how we would like certain aspects to appear. And it is only when we build the mark-up that we get to see what it looks like.

R Markdown is a variant of Markdown that is specifically designed to allow R code chunks to be included. One advantage is that we can get a ‘live’ document in which code executes and then forms part of the document. Another advantage of R Markdown is that very similar code can compile into a variety of documents, including html pages and PDFs. R Markdown also has default options set up for including a title, author, and date sections. One disadvantage is that it can take a while for a document to compile because all the code needs to run. [Tierney \(2020\)](#) is especially useful for instructions on how to achieve specific outcomes with R Markdown.

We can create a new R Markdown document within R Studio ('File' -> 'New File' -> 'R Markdown Document').

4.2.2 Essential commands

Essential markdown commands include those for emphasis, headers, lists, links, and images. A reminder of these is included in R Studio ('Help' -> 'Markdown Quick Reference').

- Emphasis: `*italic*, **bold**`
- Headers (these go on their own line with a blank line before and after): `# First level header, ## Second level header, ### Third level header`
- Unordered list, with sub-lists:

```
* Item 1
* Item 2
```

- + Item 2a
- + Item 2b
- Ordered list, with sub-lists:
 1. Item 1
 2. Item 2
 3. Item 3
 - + Item 3a
 - + Item 3b
- URLs can be added by including an address because it will auto-link: <https://www.tellingstorieswithdata.com>, or by linking some text [the address of this book](<https://www.tellingstorieswithdata.com>) results in the address of this book¹.

Once we have added some aspects, then we may want to see the actual document. To build the document click ‘Knit’.

4.2.3 R chunks

We can include code for R and many other languages in code chunks within an R Markdown document. Then when we knit the document, the code will run and be included in the document.

To create an R chunk, we start with three backticks and then within curly braces we tell R Markdown that this is an R chunk. Anything inside this chunk will be considered R code and run as such. For instance, we could load the `tidyverse` and `AER` and make a graph of the number of times a survey respondent visited the doctor in the past two weeks.

```
```{r}
library(tidyverse)
library(AER)

data("DoctorVisits", package = "AER")

DoctorVisits %>%
 ggplot(aes(x = visits)) +
 geom_histogram(stat = "count")
```
```

The output of that code is Figure 4.3.

There are various evaluation options that are available in chunks. We include these by putting a comma after `r` and then specifying any options before the closing curly brace. Helpful options include:

¹<https://www.tellingstorieswithdata.com>

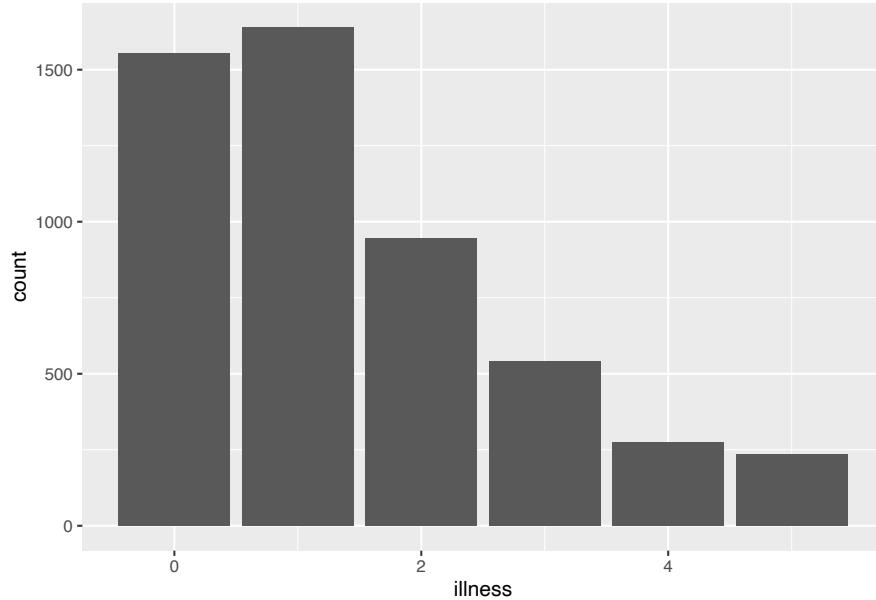


FIGURE 4.3: Number of doctor visits in the past two weeks, based on the 1977–1978 Australian Health Survey

- `echo = FALSE`: run the code and include the output, but do not print the code in the document.
- `include = FALSE`: run the code but do not output anything and do not print the code in the document.
- `eval = FALSE`: do not run the code, and hence do not include the outputs, but do print the code in the document.
- `warning = FALSE`: do not display warnings.
- `message = FALSE`: do not display messages.

For instance, we could include the output, but not the code, and suppress any warnings.

```
```{r, echo = FALSE, warning = FALSE}
library(tidyverse)
library(AER)

data("DoctorVisits", package = "AER")

DoctorVisits %>%
 ggplot(aes(x = visits)) +
 geom_histogram(stat = "count")
```
```

4.2.4 Abstracts and PDF outputs

An abstract is a short summary of the paper. In the default preamble, we can add a section for an abstract. Similarly, we can change the output from `html_document` to `pdf_document` to produce a PDF. This uses LaTeX in the background and may require the installation of supporting packages.

```
---
title: My document
author: Rohan Alexander
date: 1 January 2022
output: pdf_document
abstract: "This is my abstract."
---
```

4.2.5 References

We can include references by specifying a bib file in the preamble and then calling it within the text, as needed.

```
---
title: My document
author: Rohan Alexander
date: 1 January 2022
output: pdf_document
abstract: "This is my abstract."
bibliography: bibliography.bib
---
```

We would need to make a separate file called ‘`bibliography.bib`’ and save it next to the R Markdown file. In the bib file we need an entry for the item that is to be referenced. For instance, the citation for R can be obtained with `citation()` and this can be added to the ‘`bibliography.bib`’ file. Similarly, the citation for a package can be found by including the package name, for instance `citation('tidyverse')`. It can be helpful to use Google Scholar, or `doi2bib`, to get citations for books or articles.

```
@Manual{,
  title = {R: A Language and Environment for Statistical Computing},
  author = {{R Core Team}},
  organization = {R Foundation for Statistical Computing},
  address = {Vienna, Austria},
  year = {2021},
  url = {https://www.R-project.org/},
}
@Article{,
  title = {Welcome to the {tidyverse}},
```

```

author = {Hadley Wickham and Mara Averick and Jennifer Bryan and Winston Chang and Lucy D'Agostino Mc
year = {2019},
journal = {Journal of Open Source Software},
volume = {4},
number = {43},
pages = {1686},
doi = {10.21105/joss.01686},
}

```

We need to create a unique key that we use to refer to this item in the text. This can be anything, provided it is unique, but meaningful ones can be easier to remember, for instance ‘citeR’.

```

@Manual{citeR,
  title = {R: A Language and Environment for Statistical Computing},
  author = {{R Core Team}},
  organization = {R Foundation for Statistical Computing},
  address = {Vienna, Austria},
  year = {2021},
  url = {https://www.R-project.org/},
}

@Article{citetidyverse,
  title = {Welcome to the {tidyverse}},
  author = {Hadley Wickham and Mara Averick and Jennifer Bryan and Winston Chang and Lucy D'Agostino Mc
year = {2019},
journal = {Journal of Open Source Software},
volume = {4},
number = {43},
pages = {1686},
doi = {10.21105/joss.01686},
}

```

To cite R in the R Markdown document we include @citeR, which would put the brackets around the year, like this: [R Core Team \(2021\)](#) or [@citeR], which would put the brackets around the whole thing, like this: ([Wickham et al., 2019a](#)).

4.2.6 Cross-references

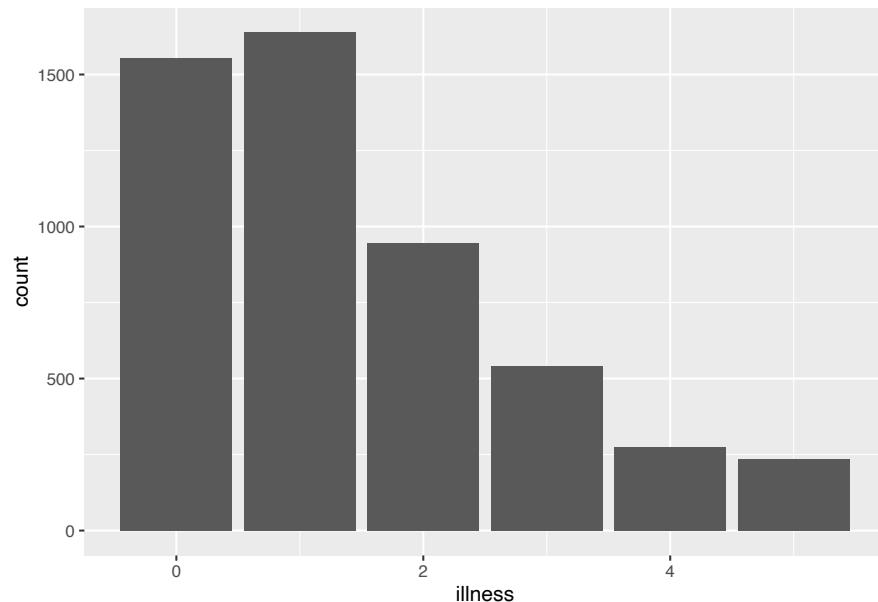
It can be useful to cross-reference figures, tables, and equations. This makes it easier to refer to them in the text. To do this for a figure we refer to the name of the R chunk that creates or contains the figure. For instance, (Figure \@ref(fig:uniquename)) will produce: (Figure 4.4) as the name of the R chunk is uniquename. We also need to add ‘fig’ in front of the chunk name so that R Markdown knows that this is a figure. We then include a ‘fig.cap’ in the R chunk that specifies a caption.

```
```{r uniquename, fig.cap = "Number of illnesses in the past two weeks, based on the 1977--1978 Au
```

```
library(tidyverse)
library(AER)

data("DoctorVisits", package = "AER")

DoctorVisits |>
 ggplot(aes(x = illness)) +
 geom_histogram(stat = "count")
```



**FIGURE 4.4:** Number of illnesses in the past two weeks, based on the 1977–1978 Australian Health Survey

We can take a similar, but slightly different, approach to cross-reference tables. For instance, (Table \@ref(tab:docvisitable)) will produce: (Table 4.1). In this case we specify ‘tab’ before the unique reference to the table, so that R Markdown knows that it is a table. For tables we need to include the caption in the main content, as a ‘caption’, rather than in a ‘fig.cap’ chunk option as is the case for figures.

**TABLE 4.1:** Number of visits to the doctor in the past two weeks, based on the 1977–1978 Australian Health Survey

visits	n
0	4141
1	782
2	174
3	30
4	24
5	9
6	12
7	12
8	5
9	1

```
DoctorVisits |>
 count(visits) |>
 knitr::kable(caption = "Number of visits to the doctor in the past two weeks, based on the 1977-1978 Australian Health Survey")
```

Finally, we can also cross-reference equations. To do that we need to add a tag (`\#eq:macroidentity`) which we then reference. For instance, use `Equation \@ref(eq:macroidentity)`. to produce Equation (4.1).

```
\begin{equation}
Y = C + I + G + (X - M) \label{eq:macroidentity}
\end{equation}
```

$$Y = C + I + G + (X - M) \quad (4.1)$$

When using cross-references, it is important that the R chunks have simple labels. In general, try to keep the names simple but unique, and if possible, avoid punctuation and stick to letters. Do not use underbars in the labels because that will cause an error.

### 4.3 R projects and file structure

We can use R Studio to create an R project. This means that we can keep all the files (data, analysis, report, etc) associated with a particular project together. A project can be created in R Studio ‘File’ -> ‘New Project’, then select ‘empty project’, name the project and decide where to save it. For

instance, a project focused on maternal mortality, may be called ‘maternal-mortality’, and might be saved within a folder of other projects. The use of R Projects is ‘the only practical convention that creates reliable, polite behavior across different computers or users and over time.’ (Bryan and Hester, 2020). Further, projects are ‘neither new, nor unique to R’, but are a well-established part of software development.

Once a project has been created, a new file with the extension ‘RProj’ will appear in that folder. As an example, of a folder with an R Project, an example R Markdown document, and an appropriate file structure is available: [https://github.com/RohanAlexander/starter\\_folder](https://github.com/RohanAlexander/starter_folder). That can be downloaded: ‘Code’ -> ‘Download ZIP’.

The main advantage of using an R Project is that we are more easily able to reference other files in a self-contained way. That means when others want to reproduce our work, they know that all the file references and structure should not need to be changed. It means that files are referenced in relation to where the ‘Rproj’ file is. For instance, instead of reading a csv from, say, “~/Documents/projects/book/data/” you can read it in from book/data/. It may be that someone else does not have a ‘projects’ folder, and so the former would not work for them, while the latter would.

The use of R projects is required to meet the minimal level of reproducibility. The use of functions such as `setwd()`, and computer-specific file paths, bind the work to your computer in a way that is not appropriate.

There are a variety of ways to set-up a folder. A variant of Wilson et al. (2017) that is often useful is shown in the example: [https://github.com/RohanAlexander/starter\\_folder](https://github.com/RohanAlexander/starter_folder). Here we have an ‘inputs’ folder that contains raw data (which should never be modified (Wilson et al., 2017)) and literature related to the project (which cannot be modified). An ‘outputs’ folder contains data that we create using R, as well as the paper that we are writing. And a ‘scripts’ folder is what modifies the raw data and saves it into ‘outputs’. Useful other aspects include a ‘README.md’ which will specify overview details about the project, and a LICENSE.

---

## 4.4 Git and GitHub

We use the combination of Git and GitHub to:

1. enhance the reproducibility of work by making it easier to share code and data;
2. make it easier to share work;
3. improve workflow by encouraging systematic approaches; and

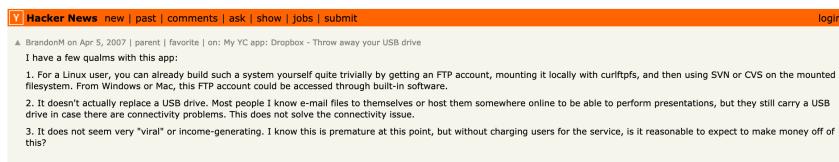
4. make it easier to work in teams.

Git is a version control system. One way one often starts doing version control is to have various versions of the one file: ‘first\\_go.R’, ‘first\\_go-fixed.R’, ‘first\\_go-fixed-with-mons-edits.R’. But this soon becomes cumbersome. One often soon turns to dates, for instance: ‘2022-01-01-analysis.R’, ‘2022-01-02-analysis.R’, ‘2022-01-03-analysis.R’, etc. While this keeps a record it can be difficult to search when we need to go back, because it can be difficult to remember the date some change was made. In any case, it quickly gets unwieldy for a project that is being regularly worked on.

Instead of this, we use Git so that we can have one version of the file, say, ‘analysis.R’ and then use Git to keep a record of the changes to that file, and a snapshot of that file at a given point in time.

We determine when Git takes that snapshot, and when we take that snapshot, we additionally include a message saying what changed between this snapshot and the last. In that way, there is only ever one version of the file, but the history can be more easily searched.

The issue is that Git was designed for software developers. As such, while it works, it can be a little ungainly for non-developers (Figure 4.5).



**FIGURE 4.5:** An infamous response to the launch of Dropbox in 2007, trivializing the use-case for Dropbox, and while this user’s approach would probably work for them, it probably would not for most folks.

Hence, GitHub, GitLab, and various other companies offer easier-to-use services that build on Git. We will introduce GitHub here because it ‘is by far the most dominant code-hosting platform’ (Eghbal, 2020, p. 21) and it is built into R Studio, but other options have advantages.

One of the challenging aspects of Git is the terminology. Folders are called ‘repos’. Saving is called a ‘commit’. One gets used to it eventually, but initially feeling confused is entirely normal.

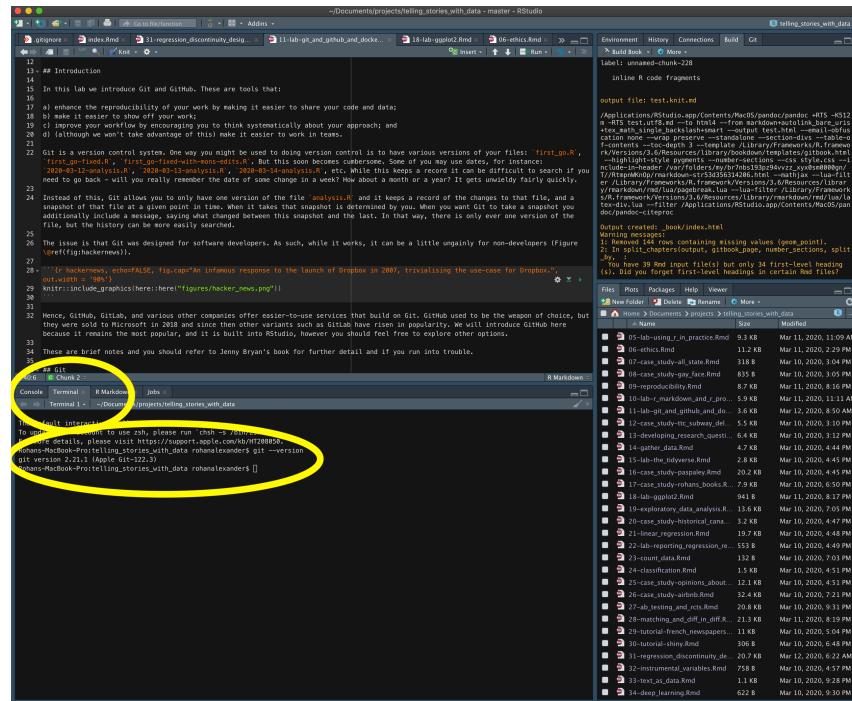
Bryan (2020) is especially useful for setting up and using Git and GitHub.

#### 4.4.1 Git

We need to git check whether Git is installed. Open R Studio, go to the Terminal, type the following, and then enter/return.

```
git --version
```

If you get a version number, then you are done (Figure 4.6).



**FIGURE 4.6:** How to access the Terminal within R Studio

If you have a Mac then Git should come pre-installed, if you have Windows then there is a chance, and if you have Linux then you probably do not need this guide. If you do not get a version number, then you need to install it. To do that you should follow the instructions specific to your operating system in Chapter 5 of [Bryan \(2020\)](#).

After we have Git, then we need to tell it our username and email. We need to do this because Git adds this information whenever we take a ‘snapshot’, or to use Git’s language, whenever we make a commit.

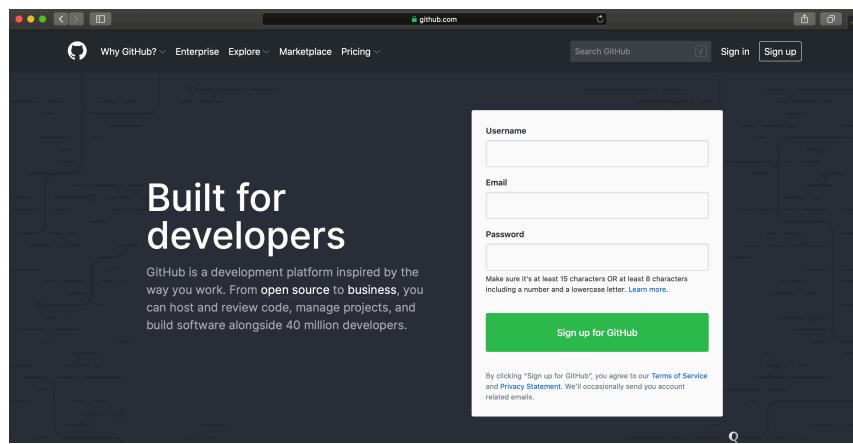
Again, within the Terminal, type the following, replacing the details with yours, and then enter/return after each line.

```
git config --global user.name 'Rohan Alexander'
git config --global user.email 'rohan.alexander@utoronto.ca'
git config --global --list
```

The details that you enter here will be public. There are various ways to hide your email address if you need to do this and GitHub provides instructions about this. Again, if you have issues, or need more detailed instructions about this step, then please see Chapter 7 of [Bryan \(2020\)](#).

#### 4.4.2 GitHub

Now that Git is set-up we need to set-up GitHub. The first step is to create an account on GitHub: <https://github.com> (Figure 4.7).



**FIGURE 4.7:** GitHub sign-up screen

We now need to make a new folder (which is called a ‘repo’ in Git). Look for a ‘+’ in the top right, and then select ‘New Repository’ (Figure 4.8).

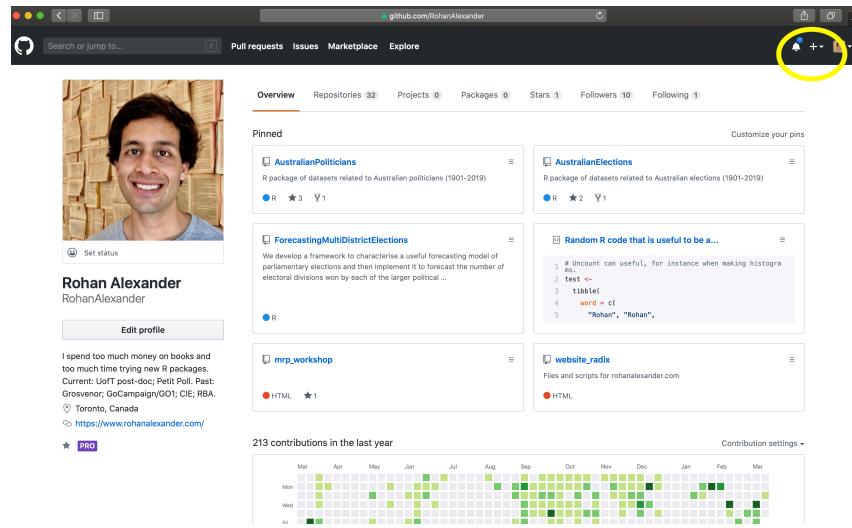
At this point we can add a sensible name for the repo. Leave it as public for now, because it can always be deleted later. And check the box to ‘Initialize this repository with a README’. Leave ‘Add .gitignore’ set to ‘None’. After that, click ‘Create repository’ (Figure 4.9).

This will take us to a screen that is fairly empty, but the details that we need are in the green ‘Clone or Download’ button, which we can copy by clicking the clipboard (Figure 4.10).

Now returning to R Studio, we open Terminal, and use `cd` to navigate to the directory where we want to create this folder. Then type the following, replacing repo details with your own, and enter/return.

```
git clone https://github.com/RohanAlexander/test.git
```

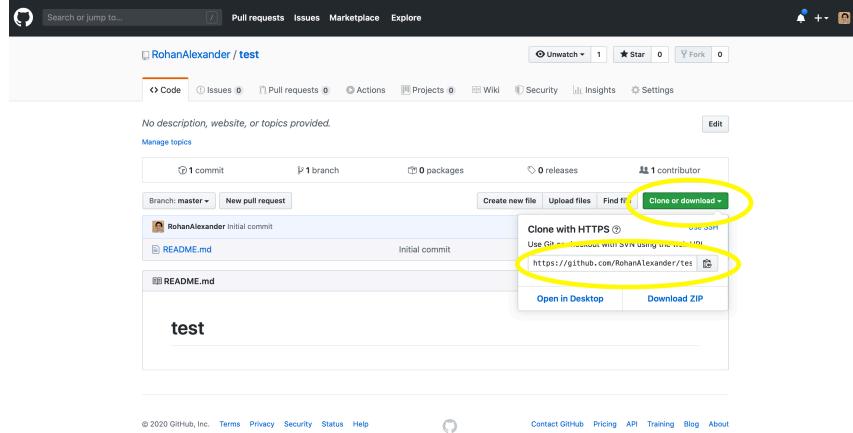
At this point, a new folder has been created and we can now use it.



**FIGURE 4.8:** Start process of creating a new repository

The screenshot shows the 'Create a new repository' form. The 'Owner' dropdown is set to 'RohanAlexander'. The 'Repository name' field contains 'test'. The 'Description (optional)' field is empty. The 'Visibility' section shows 'Public' selected (radio button is blue). Below it, a note says 'Anyone can see this repository. You choose who can commit.' The 'Private' option is also shown. Underneath, there's a note about skipping if importing an existing repository and a checked checkbox for initializing with a README. At the bottom are buttons for 'Add .gitignore: None' and 'Add a license: None', followed by a large green 'Create repository' button.

**FIGURE 4.9:** Finish creating a new repository



**FIGURE 4.10:** Get the details of your new repository

To use GitHub for a project that we are actively working on we follow a procedure:

1. The first thing to do is almost always to pull the latest changes with `git pull`. To do this, open Terminal, and navigate to the folder using `cd`. Then type `git pull` and enter/return.
2. We can then make a change to the folder, for instance, update the README, and then save it as normal.
3. Once this is done, we need to ‘add’, ‘commit’, and ‘push’.
  - As before, use `cd` to navigate to the folder, then type `git status` and enter/return to see if there are any changes (you should see some reference to the change you made).
  - Then type `git add -A` and enter/return. This adds the changes to the staging area.
  - Then type `git status` and enter/return to verify that you are happy with what was added.
  - Then type `git commit -m "Minor update to README"` and enter/return. The message should be informative about the change that was made.
  - Again, type `git status` and enter/return to check on everything.
  - Finally, type `git push` and enter/return to push everything to GitHub.

To summarize the workflow (assuming we are already in the relevant folder):

```
git pull
git status
git add -A
git status
git commit -m "Short commit message"
git status
git push
```

#### 4.4.3 Using GitHub within R Studio with the Git pane

The procedure that we went through is useful to better understand what is happening when we use Git and GitHub but can be a bit cumbersome. Usefully, GitHub is built into R Studio and so we can use the Git pane to move away from the Terminal.

Get started by creating a new repo in GitHub, as before, and copy the repo information, as before. At this point, open R Studio, and create a new project using version control ('Files' -> 'New Project' -> 'Version Control' -> 'Git', and then paste the information for the repo). Follow through the rest of the set-up by naming the project something sensible, saving it somewhere sensible, and clicking 'Open in new session', before creating the project. This will then create a new folder with an R Project which will be Git-initialized and linked to the GitHub repo.

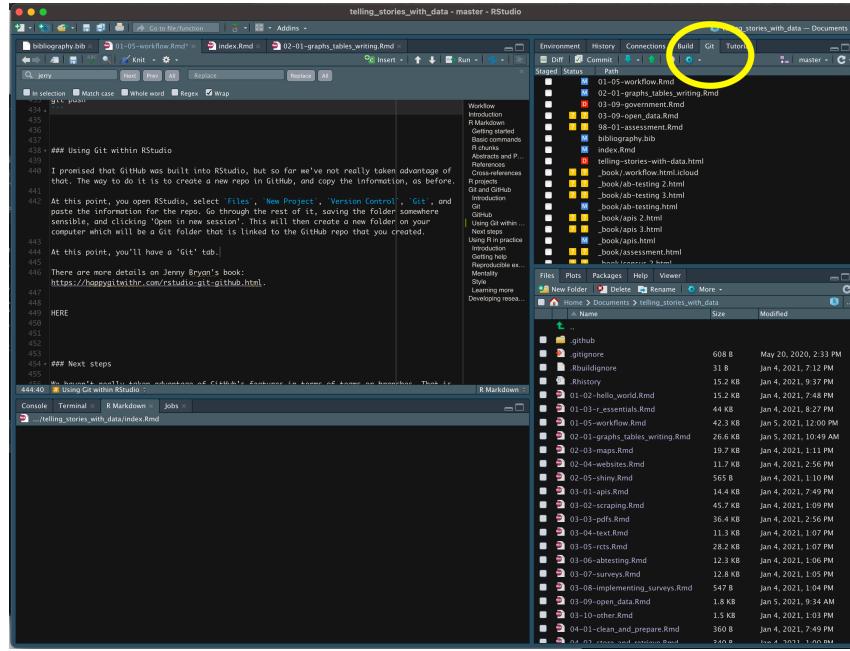
If we then open this R Project, we will have a 'Git' tab (Figure 4.11).

We can then use Git through that tab. As before, we first want to 'pull', but we can do this by clicking the blue down arrow. As before, we want to commit the files that have changes, and we do this by selecting the 'staged' checkbox against the files that we would like to commit. Then 'Commit'. Again, we want to include a message with our commit, and we do this by typing a message in the 'Commit message' box and then 'Commit'. Finally, we can 'Push'. Further details on this workflow are available in Chapter 12 of [Bryan \(2020\)](#).

#### 4.4.4 Using GitHub with usethis

We used the Git pane in R Studio to reduce the need to use the Terminal, but it did not remove the need to go to GitHub and set-up a new project. Having set-up Git and GitHub, we can further improve our workflow by using `usethis` to do much of the work ([Wickham and Bryan, 2020](#)).

After installing and loading `usethis` we need to check that we have set-up Git with `usethis::git_sitrep()`. This should print information about the user. We can use `usethis::use_git_config()` to update the username or email address. For instance,



**FIGURE 4.11:** The Git pane in R Studio

```
library(usethis)

use_git_config(user.name = "Rohan Alexander", user.email = "rohan.alexander@utoronto.ca")
```

We can then create a new R Project ('File' -> 'New Project' -> 'New Directory' -> 'New Project' -> Add a name and save it in a sensible location and click 'Open in new session').

We then use `usethis::use_git()` to initiate things. It will ask if we are happy to commit files.

Having committed, we can use `usethis::use_github()` to push to GitHub.

## 4.5 Using R in practice

### 4.5.1 Dealing with errors

When you are programming, eventually your code will break, when I say eventually, I mean like probably 10 or 20 times a day.

Gelfand (2021)

---

Everyone who uses R, or any programming language for that matter, has trouble find them at some point. This is normal. Programming is hard and everyone struggles sometimes. At some point code will not run or will throw an error. This is normal, and it happens to everyone. Everyone gets frustrated.

To move forward we develop strategies to work through the issues:

1. If you are get an error message, then sometimes it will be useful. Try to read it carefully to see if there is anything of use in it.
2. Try to search, say on Google, for the error message. It can be useful to include ‘tidyverse’ or ‘in R’ in the search to help make the results more appropriate. Sometimes Stack Overflow results can be useful.
3. Look at the help file for the function, by putting ‘?’ before the function, for instance, `?pivot_wider()`. A common issue is to use a slightly incorrect argument name or format, such as accidentally including a string instead of an object name.
4. Look at where the error is happening and remove code until the error is resolved, and then slowly add code back again.
5. Check the class of the object, with `class()`, for instance, `class(data_set$data_column)`. Ensuring that it is what it expected.
6. Restart R (‘Session’ -> ‘Restart R and Clear Output’) and load everything again.
7. Restart the computer.
8. Search for what you are trying to do, rather than the error, being sure to include ‘tidyverse’ or ‘in R’ in the search to help make the results more appropriate. For instance, ‘save PDF of graph in R made using ggplot’. Sometimes there are relevant blog posts or Stack Overflow answers that will help.
9. Making a small, self-contained, reproducible example ‘reprex’ to see if the issue can be isolated and to enable others to help.

More generally, while this is rarely possible to do, it is almost always helpful to take a break and come back the next day.

#### 4.5.2 Reproducible examples

---

No one can advise or help you—no one. There is only one thing you should do. Go into yourself.

Rilke (1929)

---

Asking for help is a skill like any other. We get better at it with practice. It is important to try not to say ‘this doesn’t work’, ‘I tried everything’, ‘your code does not work’, or ‘here is the error message, what do I do?’ In general, it is not possible to help based on these comments, because there are too many possible issues. You need to make it easy for others to help you. This involves a few steps.

1. Provide a small, self-contained, example of your data, and code, and detail what is going wrong.
2. Document what you have tried so far, including which Stack Overflow and R Studio Community pages have you looked at, and why are they not quite what you are after?
3. Be clear about the outcome that you would like.

Begin by creating a minimal REPRoducible EXample, a ‘reprex’. This is code that contains what is needed to reproduce the error, but only what is needed. This means that the code is likely a smaller, simpler, version that nonetheless reproduces the error.

Sometimes this process enables one to solve the problem. If it does not, then it gives someone else a fighting chance of being able to help. It is important to recognize that there is almost no chance that you have got a problem that someone has not addressed before. It is more likely that the main difficulty is in trying to communicate what you are trying to do and what is happening, in a way that allows others to recognize both. Developing tenacity is important.

To develop reproducible examples, the `reprex` package (Bryan et al., 2019) is especially useful. To use it we:

1. Load the `reprex` package: `library(reprex)`.
2. Highlight and copy the code that is giving issues.
3. Run `reprex()` in the Console.

If the code is self-contained, then it will preview in the Viewer. If it is not, then it will error, and the code needs to be re-written so that it is self-contained.

If you need data to reproduce the error, then you should use data that is built into R. There are a large number of datasets that are built into R and can be seen using `library(help = "datasets")`. But if possible, you should use a common option such as ‘mtcars’ or ‘faithful’.

### 4.5.3 Mentality

---

(Y)ou are a real, valid, *competent* user and programmer no matter what IDE you develop in or what tools you use to make your work work for you

(L)et's break down the gates, there's enough room for everyone

Sharla Gelfand, 10 March 2020<sup>2</sup>.

---

If you write code, then you are a programmer regardless of how you do it, what you are using it for, or who you are. But there are a few traits that one tends to notice great programmers have in common.

- Focused: Often having an aim to ‘learn R’ or something similar tends to be problematic, because there is no real end point to that. It tends to be more efficient to have smaller, more specific goals, such as ‘make a histogram about the 2019 Canadian Election with ggplot’. This is something that can be focused on and achieved in a few hours. The issue with goals that are more nebulous, such as ‘I want to learn R’, is that it becomes easy to get lost on tangents, much more difficult to get help. This can be demoralizing and lead to folks quitting too early.
- Curious: It is almost always useful to have a go. In general, the worst that happens is that you waste your time. You can rarely break something irreparably with code. If you want to know what happens if you pass a ‘vector’ instead of a ‘dataframe’ to `ggplot()` then try it.
- Pragmatic: At the same time, it can be useful to stick within reasonable bounds, and make one small change each time. For instance, say you want to run some regressions, and are curious about the possibility of using the `tidymodels` package (Kuhn and Wickham, 2020) instead of `lm()`. A pragmatic way to proceed is to use one aspect from the `tidymodels` package initially and then make another change next time.
- Tenacious: Again, this is a balancing act. There are always unexpected problems and issues with every project. On the one hand, persevering despite these is a good tendency. But on the other hand, sometimes one does need to be prepared to give up on something if it does not seem like a break-through is possible. Here mentors can be useful as they tend to be a better judge of what is reasonable. It is also where appropriate planning is useful.
- Planned: It is almost always useful to excessively plan what you are going to do. For instance, you may want to make a histogram of the 2019 Canadian

<sup>2</sup><https://twitter.com/sharlagelfand/status/1237365576701542400>

Election. You should plan the steps that are needed and even to sketch out how each step might be implemented. For instance, the first step is to get the data. What packages might be useful? Where might the data be? What is the back-up plan if the data do not exist there?

- Done is better than perfect: We all have various perfectionist tendencies to a certain extent, but it can be useful to initially try to turn them off to a certain extent. In the first instance, try to write code that works, especially in the early days. You can always come back and improve aspects of it. But it is important to actually ship. Ugly code that gets the job done, is better than beautiful code that is never finished.

#### 4.5.4 Code comments and style

Code must be commented ([Lee, 2018](#)). Comments should focus on why certain code was written, (and to a lesser extent, why a common option is not selected).

There is no one way to write code, especially in R. However, there are some general guidelines that will make it easier for you even if you are just working on your own. It is important to recognize that most projects will evolve over time, and one purpose served by code comments are as ‘[m]essages left for your future self (or near-future others) [that] help retrace and justify your decisions’ ([Bowers, 2011](#)).

Comments in R can be added by including the # symbol. We do not have to put a comment at the start of the line, it can be midway through. In general, we do not need to comment what every aspect of your code is doing but we should comment parts that are not obvious. For instance, if we read in some value then we may like to comment where it is coming from.

You should comment why you are doing something ([Wickham, 2021c](#)). What are you trying to achieve?

You must comment to explain weird things. Like if you are removing some specific row, say row 27, then why are you removing that row? It may seem obvious in the moment, but future-you in six months will not remember.

You should break your code into sections. For instance, setting up the workspace, reading in datasets, manipulating and cleaning the dataset, analyzing the datasets, and finally producing tables and figures. Each of these should be separated with comments explaining what is going on, and sometimes into separate files, depending on the length.

Additionally, at the top of each file it is important to basic information, such as the purpose of the file, and pre-requisites or dependencies, the date, the author and contact information, and finally and red-flags or todos.

At the very least every R script needs a preamble and a clear demarcation of sections.

```
Preamble
Purpose: Brief sentence about what this script does
Author: Your name
Data: The date it was written
Contact: Add your email
License: Think about how your code may be used
Pre-requisites:
- Maybe you need some data or some other script to have been run?

Workspace setup
do not keep the install.packages line - comment out if need be
Load libraries
library(tidyverse)

Read in the raw data.
raw_data <- readr::read_csv("inputs/data/raw_data.csv")

Next section
...
```

---

## 4.6 Exercises and tutorial

### 4.6.1 Exercises

1. According to Alexander (2019c) research is reproducible if (pick one)?
  - a. It is published in peer-reviewed journals.
  - b. All of the materials used in the study are provided.
  - c. It can be reproduced exactly without the authors providing materials.
  - d. It can be reproduced exactly, given all the materials used in the study.
2. According to the timeline of Gelman (2016), a) when did Paul Meehl identify various issues; and b) when did null hypothesis significance testing (NHST) become controversial (pick one)?
  - a. 1970s-1980s; 1990s-2000.
  - b. 1960s-1970s; 1980s-1990.
  - c. 1970s-1980s; 1980s-1990.

- d. 1960s-1970s; 1990s-2000.
3. Which of the following are components of the project layout recommended by Wilson et al. (2017) (select all that apply)?
  - a. requirements.txt
  - b. doc
  - c. data
  - d. LICENSE
  - e. CITATION
  - f. README
  - g. src
  - h. results
4. Based on Alexander (2021) please write a paragraph about some of the barriers you overcame, or still face, with regard to sharing code that you wrote.
5. According to Gelfand (2021), what is the key part of ‘If you need help getting unstuck, the first step is to create a reprex, or reproducible example. The goal of a reprex is to package your problematic code in such a way that other people can run it and feel your pain. Then, hopefully, they can provide a solution and put you out of your misery.’ (pick one)?
  - a. package your problematic code
  - b. other people can run it and feel your pain
  - c. the first step is to create a reprex
  - d. they can provide a solution and put you out of your misery
6. According to Gelfand (2021), what are the three key aspects of a reprex (select all that apply)?
  - a. data
  - b. only the libraries that are necessary and all the libraries that are necessary
  - c. relevant code and only relevant code
7. According to Wickham (2021c) for naming files, how would these files ‘00\_get\_data.R’, ‘get data.R’ be classified (pick one)?
  - a. bad; bad.
  - b. good; bad.
  - c. bad; good.
  - d. good; good.
8. Which of the following would result in bold text in R Markdown (pick one)?
  - a. **\*\*bold\*\***
  - b. **##bold##**
  - c. **\*bold\***
  - d. **#bold#**
9. Which option would hide the warnings in a R Markdown R chunk (pick one)?
  - a. echo = FALSE

- b. `include = FALSE`
  - c. `eval = FALSE`
  - d. `warning = FALSE`
  - e. `message = FALSE`
10. Which options would run the R code chunk, but not display the code in a R Markdown R chunk (pick one)?
- a. `echo = FALSE`
  - b. `include = FALSE`
  - c. `eval = FALSE`
  - d. `warning = FALSE`
  - e. `message = FALSE`
11. Why are R Projects important (select all that apply)?
- a. They help with reproducibility.
  - b. They make it easier to share code.
  - c. They make your workspace more organized.
  - d. They ensure reproducibility.
12. Please discuss a circumstance in which an R Project would be useful.
13. Consider this sequence: ‘`git pull`, `git status`, \_\_\_\_\_, `git status`, `git commit -m "My message"`, `git push`’. What is the missing step (pick one)?
- a. `git add -A`.
  - b. `git status`.
  - c. `git pull`.
  - d. `git push`.
14. Assuming the libraries and datasets have been loaded, what is the mistake in this code: `DoctorVisits |> select("visits")` (pick one)?
- a. “`visits`”
  - b. `DoctorVisits`
  - c. `select`
  - d. `|>`
15. What is a reprex and why is it important to be able to make one (select all that apply)?
- a. A reproducible example that enables your error to be reproduced.
  - b. A reproducible example that helps others help you.
  - c. A reproducible example during the construction of which you may solve your own problem.
  - d. A reproducible example that demonstrates you have actually tried to help yourself.
16. The following code produces an error. Please use `reprex` (Bryan et al., 2019) to build a reproducible example that you could use to get help with it, and submit the reprex.

```

library(tidyverse)

oecd_gdp <-
 read_csv("https://stats.oecd.org/sdmx-json/data/DP_LIVE/.QGDP.../OECD?contentType=csv&detail=cod")

head(oecd_gdp)

library(forcats)
library(dplyr)

oecd_gdp_most_recent <-
 oecd_gdp |>
 filter(TIME == "2021-Q3",
 SUBJECT == "TOT",
 LOCATION %in% c("AUS", "CAN", "CHL", "DEU", "GBR",
 "IDN", "ESP", "NZL", "USA", "ZAF"),
 MEASURE == "PC_CHGPy") |>
 mutate(european = if_else(LOCATION %in% c("DEU", "GBR", "ESP"),
 "European",
 "Not european"),
 hemisphere = if_else(LOCATION %in% c("CAN", "DEU", "GBR", "ESP", "USA"),
 "Northern Hemisphere",
 "Southern Hemisphere"),
)
)

library(ggplot)
library(patchwork)

oecd_gdp_most_recent |>
 ggplot(mapping = aes(x = LOCATION, y = Value)) |>
 geom_bar(stat="identity")

```

#### 4.6.2 Tutorial

Please put together a small R Markdown file that downloads a dataset using `opendatatoronto`, cleans it, and makes a graph. Then exchange it with someone else. Ask them to both read the code and to run it, and to then provide you with feedback about both aspects. Write a page or two of single-spaced content, about the comments that you received and changes that you could make going forward.

### **4.6.3 Paper**

At about this point, Paper 1 (Appendix 7.5.3) would be appropriate.



---

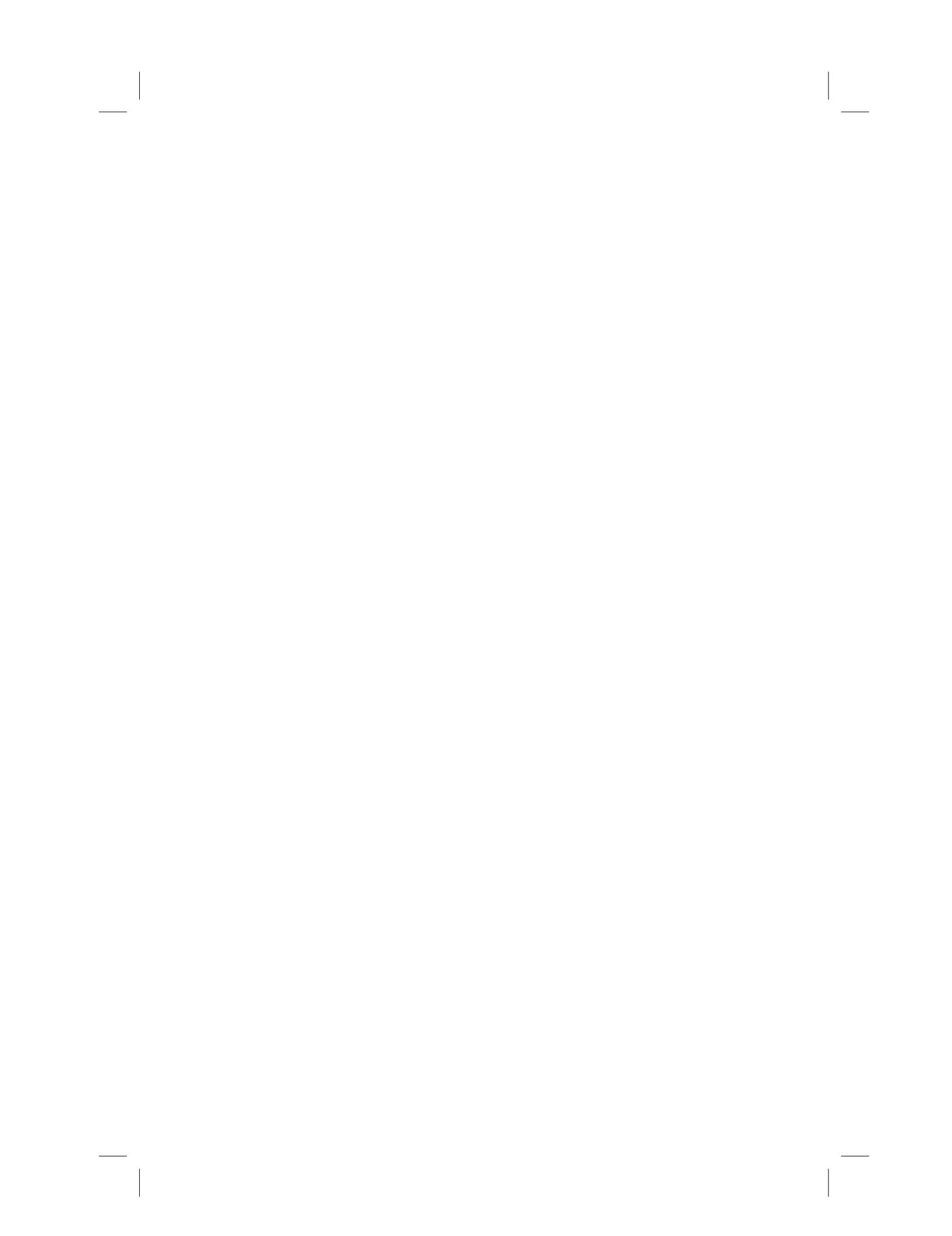
---

## **Part II**

# **Communication**

---

---



# 5

---

## *On writing*

---

### Required material

---

If you want to be a writer, you must do two things above all others: read a lot and write a lot. There's no way around these two things that I'm aware of, no shortcut.

King (2000, p. 145)

---

- Read *On Writing Well*, (any edition is fine) (Zinsser, 1976).
- Read *Publication, publication*, (King, 2006)
- Read two of the following well-written quantitative papers:
  - *Asset prices in an exchange economy*, (Lucas Jr, 1978).
  - *Individuals, institutions, and innovation in the debates of the French Revolution*, (Barron et al., 2018).
  - *Modeling: optimal marathon performance on the basis of physiological factors*, (Joyner, 1991).
  - *On reproducible econometric research*, (Koenker and Zeileis, 2009).
  - *Prevented mortality and greenhouse gas emissions from historical and projected nuclear power*, (Kharecha and Hansen, 2013).
  - *Sample selection bias as a specification error*, (Heckman, 1979).
  - *Seeing like a market*, (Fourcade and Healy, 2017).
  - *Simpson's paradox and the hot hand in basketball*, (Wardrop, 1995).
  - *Smoking and carcinoma of the lung*, (Doll and Hill, 1950).
  - *Some studies in machine learning using the game of checkers*, (Samuel, 1959).
  - *Statistical methods for assessing agreement between two methods of clinical measurement*, (Bland and Altman, 1986).
  - *The mundanity of excellence: An ethnographic report on stratification and Olympic swimmers*, (Chambliss, 1989).
  - *The probable error of a mean*, (Student, 1908).
- Read two of the following articles from *The New Yorker*:
  - *Funny Like a Guy*, Tad Friend

- *Going the Distance*, David Remnick
- *Happy Feet*, Alexandra Jacobs
- *Levels of the Game*, John McPhee
- *Reporting from Hiroshima*, John Hersey
- *The Catastrophist*, Elizabeth Kolbert
- *The Quiet German*, George Packer
- Read two of the following articles from miscellaneous publications:
  - *Blades of Glory*, Holly Anderson
  - *Born to Run*, Walt Harrington
  - *Dropped*, Jason Fagone
  - *Federer as Religious Experience*, David Foster Wallace
  - *Generation Why?*, Zadie Smith
  - *One hundred years of arm bars*, David Samuels
  - *Out in the Great Alone*, Brian Phillips
  - *Pearls Before Breakfast*, Gene Weingarten
  - *The Cult of ‘Jurassic Park’*, Bryan Curtis
  - *The House that Hova Built*, Zadie Smith
  - *The Re-Education of Chris Copeland*, Flinder Boyd
  - *The Sea of Crisis*, Brian Phillips

### Key concepts and skills

- To get better at writing, write, ideally every day.
- Write for the reader.
- Have one message that you want to communicate.
- Get to a first draft as quickly as possible.
- Rewrite brutally.
- Remove as many words as possible.

---

## 5.1 Introduction

---

[T]he duty of a scientist is not only to find new things, but to communicate them successfully in at least three forms: 1) Writing papers and books. 2) Prepared public talks. 3) Impromptu talks.

Hamming (1996, p. 65)

---

---

People who need to write: founders, VCs, lawyers, software engineers, designers, painters, data scientists, musicians, filmmakers, creative directors, physical trainers, teachers, writers. Learn to write.

Sahil Lavingia, 3 February 2020.  

---

---

Writing well has done just as much for me as knowing how to code. I'd add that if you're intimidated by writing, start a blog and write often about something you're interested in. You'll get better. At least that's what I've done for the past 10 years. :)

Vicki Boykis, 3 February 2020.  

---

We need to write in order to tell our stories. Writing allows us to communicate efficiently. It is also a way to work out what we believe and allows us to get feedback on our ideas. Effective papers are tightly written and well-organized, which makes the story flow and easy to follow. Proper sentence structure, spelling, vocabulary, and grammar are important because they remove distractions and enable each point to be clearly articulated. Effective papers demonstrate understanding of the topic by confidently using relevant terms and techniques and considering issues without being overly verbose. Graphs, tables, and references are used to enhance both the story and its credibility.

This chapter is about writing. By the end of it, you will have a better idea of how to write short, detailed, quantitative papers that communicate what you want them to, and do not waste the reader's time. We write for the reader, not for ourselves. Specifically, we write to be useful to the reader, where '[u]seful writing tells people something true and important that they didn't already know, and tells them as unequivocally as possible' (Graham, 2020). That said, the greatest benefit of writing nonetheless often accrues to the writer, even when we write for our audience. This is because the process of writing is a way to work out what we think and how we came to believe it.

---

The way to do a piece of writing is three or four times over, never

once. For me, the hardest part comes first, getting something—anything—out in front of me. Sometimes in a nervous frenzy I just fling words as if I were flinging mud at a wall. Blurt out, heave out, babble out something—anything—as a first draft. With that, you have achieved a sort of nucleus. Then, as you work it over and alter it, you begin to shape sentences that score higher with the ear and the eye. Edit it again—top to bottom. The chances are that about now you'll be seeing something that you are sort of eager for others to see. And all that takes times. What I have left out is the interstitial time. You finish that first awful blurting, and then you put the thing aside. You get in your car and drive home. On the way, your mind is still knitting at the words. You think of a better way to say something, a good phrase to correct a certain problem. Without that drafted version—if it did not exist—you obviously would not be thinking of things that would improve it. In short, you may be actually writing only two or three hours a day, but your mind, in one way or another, is working on it twenty-four hours a day—yes, while you sleep—but only if some sort of draft of earlier version already exists. Until it exists, writing has not really begun.

McPhee (2017, p. 159)

---

The process of writing is a process of re-writing. And the critical task is to get to a first draft as quickly as possible. A complete first draft of a five-to-ten-page quantitative paper can be done in a day. Until that complete first draft exists, it is useful to try to not to delete or even revise anything that was written, regardless of how bad it may seem. Just write.

One of the most intimidating things in the world is a blank page, and we deal with this by immediately adding headings such as: ‘Introduction’, ‘Data’, ‘Model’, ‘Results’, and ‘Discussion’. And then add fields in the top matter for the various bits and pieces that are needed, such as ‘title’, ‘date’, ‘author’ and ‘abstract’. This creates a generic outline, and its role is akin to placing on the counter, the ingredients that we will use to prepare dinner (McPhee, 2017).

Having established this generic outline, we need to develop an understanding of what we are exploring through developing a research question. In theory, we develop a research question, answer it, and then we do all the writing; but that rarely actually happens (Franklin, 2005). Instead, we typically have some idea of the question, and our answer, and these become less vague as we write. This is because it is through the process of writing that we refine our thinking (King, 2000, p. 131). Having put down some thoughts about the research question, we can start to add dot points in each of the sections, adding

sub-sections, with informative sub-headings as needed. We then go back and expand those dot points into paragraphs.

While writing the first draft it is important to ignore the feeling that one is not good enough, or that it is impossible. Just write. We need words on paper, even if they are bad, and the first draft is when we accomplish this. Remove all distractions and just write. Perfectionism is the enemy, and should be set aside. Sometimes this can be accomplished by getting up very early to write, or by creating a deadline, or with a glass or two of wine. One friend puts her baby to sleep to the sound of her typing, with the result being that she must keep typing otherwise the baby will wake up. Creating a sense of urgency can be useful and rather than adding proper citations as we go, which could slow us down, just add something like '[TODO: CITE R HERE]'. Do similar with graphs and tables. That is, include textual descriptions such as '[TODO: ADD GRAPH THAT SHOWS EACH COUNTRY OVER TIME HERE]' instead of actual graphs and tables. Focus on adding content, even if it is poorly written, or not ideal. When this is all done, a first draft exists!

This first draft will be bad. But it is by writing a bad first draft that we can get to a good second draft, a great third draft, and eventually excellence (Lamott, 1994, p. 20). That first draft will be too long, it will not make sense, it will contain claims that cannot be supported, and some claims that should not be. Having focused on adding content while writing the first draft, to turn that into a second draft, we use the 'delete' key extensively, as well as 'cut' and 'paste'. Printing out the paper and using a red pen to move or remove is especially helpful. The process of going from a first draft to a second draft is best done in one sitting, to help with flow and consistency of the story. One aspect of this first re-write is enhancing the story that we want to tell. And another aspect is taking out everything that is not the story (King, 2000, p. 57).

As we go through what was written in each of the sections, we try to bring some sense to it, with special consideration to how it supports our story. This revision process is the essence of writing (McPhee, 2017, p. 160). We should also fix the references, and add the real graphs and tables. As part of this re-writing process, the paper's central message tends to develop, and our answers to the research questions tend to become clearer. At this point, aspects such as the introduction can be returned to and, finally, the abstract. Typos and other issues affect the credibility of the work, and so it is important that these are fixed as part of the second draft.

We now have a paper that is sensible. The job is to now make it brilliant. Print it out again, and again go through it on paper. It is especially important to brutally remove everything that does not contribute to the story. At about this stage, we may be starting to get too close to the paper. We write for our reader, and so this is a great opportunity to give it to someone else for their comments. We ask them for feedback that enables us to better understand

the weak parts of the story. After addressing these, it can be helpful to go through the paper once more, this time reading it aloud. A paper tends to never be ‘done’ and it is more that at a certain point we either run out of time or become sick of the sight of it.

---

## 5.2 Developing research questions

Both qualitative and quantitative approaches have their place, but here we focus on quantitative approaches. Qualitative research is important as well, and often the most interesting work has a little of both. When conducting quantitative analysis, we are subject to issues such as data quality, scales, measurement, and sources. We are often especially interested in trying to tease out causality. Regardless, we are trying to learn something about the world. Our research questions need to take this all into account.

Broadly, there are two ways to go about research:

- 1) data-first; or
- 2) question-first.

### 5.2.1 Data-first

When being data-first, the main issue is working out the questions that can be reasonably answered with the available data. When deciding what these are, it is useful to consider:

- 1) Theory: Is there a reasonable expectation that there is something causal that could be determined? For instance, if the question involves charting the stock market, then it might be better to consider haruspex because at least that way we would have something to eat. Questions usually need to have some plausible theoretical underpinning to help avoid spurious relationships.
- 2) Importance: There are plenty of trivial questions that can be answered, but it is important to not waste our time or that of the reader. Having an important question can also help with motivation when we find ourselves in, say, the fourth straight week of cleaning data and de-bugging code. It can also make it easier to attract talented employees and funding. That said, there is a balance that is needed. But it is important that the question has a decent chance of being answered. And so attacking a generational-defining question might be best broken up into smaller chunks.
- 3) Availability: Is there a reasonable expectation of additional data

- being available in the future? This could allow us to answer related questions and turn this one paper into a research agenda.
- 4) Iteration: Is this something that could be run multiple times, or is it a once-off analysis? If it is the former, then it becomes possible to start answering specific research questions and then iterate. But if we can only get access to the data once then we need to think about broader questions.

There's a saying, sometimes attributed to Xiao-Li Meng that all of statistics is a missing data problem. And so paradoxically, another way to ask data-first questions to think about which data we do not have. For instance, returning to the neonatal and maternal mortality examples discussed in, respectively, Chapters 1 and 2, the fundamental problem is that we do not have perfect and complete data about cause of death. If we did, then we could count the number of relevant deaths. Having established the missing data problem, we can take a data-driven approach by looking at the data we do have, and then ask research questions that speak to the extent that we can use that to approximate our hypothetical perfect and complete dataset.

### 5.2.2 Question-first

When trying to be question-first, there is the inverse different issue of being concerned about data availability. The 'FINER framework' is used in medicine to help guide the development of research questions. It recommends asking questions that are: Feasible, Interesting, Novel, Ethical, and Relevant (Hulley, 2007). Farrugia et al. (2010) builds on FINER with PICOT, which recommends additional considerations: Population, Intervention, Comparison group, Outcome of interest, and Time. It can feel overwhelming trying to write out a question. One way to go about it is to ask a very specific question. Another is to decide whether we are interested in descriptive, predictive, inferential, or causal analysis.

These then lead to different types of questions, for instance, descriptive analysis: 'What does  $x$  look like?'; predictive analysis: 'What will happen to  $x$ ?'; inferential: 'How can we explain  $x$ ?'; and causal: 'What impact does  $x$  have on  $y$ ?'. Each of these have a role to play.

Often time will be constrained, possibly in interesting ways and these can guide the specifics of the research question. If we are interested in the effect of Trump's tweets on the stock market, then that can be done just by looking at the minutes (milliseconds?) after he tweets. But what if we are interested in the effect of a cancer drug on long term outcomes? If the effect takes 20 years, then we must either wait a while, or we need to look at people who were treated in 2000, but then we have selection effects and different circumstances to if we give the drug today. Often the only reasonable thing to do is to build a statistical model, but then we need adequate sample sizes, etc.

When answering questions usually, the creation of a counterfactual is crucial. Briefly, a counterfactual is an if-then statement in which the ‘if’ is false. Consider the example of Humpty Dumpty from Lewis Carroll’s *Through the Looking-Glass* (Carroll, 1871).

---

‘What tremendously easy riddles you ask!’ Humpty Dumpty growled out. ‘Of course I don’t think so! Why, if ever I did fall off—which there’s no chance of—but if I did—’ Here he pursed his lips and looked so solemn and grand that Alice could hardly help laughing. ‘If I did fall,’ he went on, ‘The King has promised me—with his very own mouth-to-to—’

---

Humpty is satisfied with what would happen if he were to fall off, even though he is similarly satisfied that this would never happen. It is this comparison group that often determines the answer to a question. For instance, consider the effect of VO<sub>2</sub> max on the outcome of bike race. If we compare over the general population then it is an important variable, but if we only compare over elite athletes, then it is less important, because of selection.

---

### 5.3 Writing

---

I had not indeed published anything before I commenced “The Professor”, but in many a crude effort, destroyed almost as soon as composed, I had got over any such taste as I might once have had for ornamented and redundant composition, and come to prefer what was plain and homely.

*The Professor* (Brontë, 1857).

---

We discuss the following components: title, abstract, introduction, data, results, discussion, figures, tables, equations, and technical terms. Throughout

all sections of a paper it is important that we are as brief and specific as possible.

### 5.3.1 Title

A title is the first opportunity that we have to engage our reader in our story. Ideally, we are able to tell our reader exactly what we found. Effective titles are critical because otherwise papers will be ignored by readers. While a title does not have to be ‘cute’, it does need to be effective. This means it needs to make the story clear.

One example of a title that is good enough is ‘On the 2016 Brexit referendum’. This title is useful because the reader at least knows what the paper will be about. But it is not particularly informative or enticing. A slightly better variant could be ‘On the ‘Vote Leave’ outcome in the 2016 Brexit referendum’. This variant adds specifically which is particularly informative. Finally, another variant would be ‘Vote Leave outperforms in rural areas in the 2016 Brexit referendum: Evidence from a Bayesian hierarchical model’. Here the reader knows the approach of the paper and also the main take-away.

We will consider a few examples of particularly effective titles. [Hug et al. \(2019\)](#) uses ‘National, regional, and global levels and trends in neonatal mortality between 1990 and 2017, with scenario-based projections to 2030: a systematic analysis’. Here it is clear what the paper is about and the methods that are used. [Alexander and Alexander \(2021\)](#) uses ‘The Increased Effect of Elections and Changing Prime Ministers on Topics Discussed in the Australian Federal Parliament between 1901 and 2018’. While the method used in that paper is not clear from the title, the main finding is, along with a good deal of information about what the content will be. And finally, [Alexander et al. \(2018\)](#) uses ‘Trends in Black and White Opioid Mortality in the United States, 1979–2015’.

A title is often among the last aspects of a paper to be finalized. While getting through the first draft, we would typically just use a working title that is good enough to get the job done. We then refine it over the course of redrafting. The title needs to reflect the final story of the paper, and this is not usually something that we know at the start. We are interested in striking a balance between getting our reader interested enough to read the paper, and conveying enough of the content so as to be useful ([Hayot, 2014](#)). We can think here of classic books, such as Macaulay’s *History of England from the Accession of James the Second*, or Churchill’s *A History of the English-Speaking Peoples*. Both are clear about what the content is, and, for their target audience, spark interest.

One specific approach is the form: ‘Exciting content: Specific content’, for instance, ‘Returning to their roots: Examining the performance of ‘Vote Leave’ in the 2016 Brexit referendum’. [Kennedy and Gelman \(2020\)](#) provides a par-

ticular nice example of this approach with ‘Know your population and know your model: Using model-based regression and poststratification to generalize findings beyond the observed sample’, as does [Craiu \(2019\)](#) with ‘The Hiring Gambit: In Search of the Twofer Data Scientist’. A close variant of this is ‘A question? And an answer’. For instance, [Cahill et al. \(2020\)](#) with ‘What increase in modern contraceptive use is needed in FP2020 countries to reach 75% demand satisfied by 2030? An assessment using the Accelerated Transition Method and Family Planning Estimation Model’. As one gains experience with this variant, it becomes possible to know when it is appropriate to drop the answer part yet remain effective, such as [Briggs \(2021\)](#) with ‘Why Does Aid Not Target the Poorest?’. Another specific approach is ‘Specific content then broad content’ or inversely. For instance ‘Rurality, elites, and support for ‘Vote Leave’ in the 2016 Brexit referendum’ or ‘Support for ‘Vote Leave’ in the 2016 Brexit referendum, rurality and elites. This approach is used by [Tolley and Paquet \(2021\)](#) with ‘Gender, municipal party politics, and Montreal’s first woman mayor’.

### 5.3.2 Abstract

For a five-to-ten-page paper, a good abstract is a three to five sentence paragraph. For a longer paper the abstract can be slightly longer. The abstract needs to specify the story of the paper, and the objective of an abstract is to convey what was done and why it matters. To do this an abstract typically touches on the context of the work, its objectives, approach, and findings.

More specifically, a good recipe for an abstract is: first sentence: specify the general area of the paper and encourage the reader; second sentence: specify the dataset and methods at a general level; third sentence: specify the headline result; and a fourth sentence about implications.

We see this pattern in a variety of abstracts. For instance, [Tolley and Paquet \(2021\)](#) draw in the reader with their first sentence by mentioning the election of the first woman mayor in 400 years. The second sentence is clear about what is done in the paper. The third paper tells the reader how it is done i.e. a survey. And the fourth sentence adds some detail. The fifth and final sentence makes the main take-away from the paper clear.

---

In 2017, Montreal elected Valérie Plante, the first woman mayor in the city’s 400-year history. Using this election as a case study, we show how gender did and did not influence the outcome. A survey of Montreal electors suggests that gender was not a salient factor in vote choice. Although gender did not matter much for voters, it did shape the organization of the campaign

and party. We argue that Plante's victory can be explained in part by a strategy that showcased a less leader-centric party and a degendered campaign that helped counteract stereotypes about women's unsuitability for positions of political leadership.

---

Similarly, Beauregard and Sheppard (2021) make broader environment clear within the first two sentences, and the specific contribution of this paper to that environment. The third and fourth sentences makes the data source clear and also the main findings. The fifth and sixth sentences add specificity here that would be of interest to likely readers of this abstract i.e. academic political science experts. And then the final sentence makes it clear the position of the authors.

---

Previous research on support for gender quotas focuses on attitudes toward gender equality and government intervention as explanations. We argue the role of attitudes toward women in understanding support for policies aiming to increase the presence of women in politics is ambivalent—both hostile and benevolent forms of sexism contribute in understanding support, albeit in different ways. Using original data from a survey conducted on a probability-based sample of Australian respondents, our findings demonstrate that hostile sexists are more likely to oppose increasing of women's presence in politics through the adoption of gender quotas. Benevolent sexists, on the other hand, are more likely to support these policies than respondents exhibiting low levels of benevolent sexism. We argue this is because benevolent sexism holds that women are pure and need protection; they do not have what it takes to succeed in politics without the assistance of quotas. Finally, we show that while women are more likely to support quotas, ambivalent sexism has the same relationship with support among both women and men. These findings suggest that aggregate levels of public support for gender quotas do not necessarily represent greater acceptance of gender equality generally.

---

And finally, Briggs (2021) begins with a claim that seems unquestionably true. In the second sentence he then claims to have found that it is false. The third sentence specifies the extent of this claim, and the fourth sentence details

how he comes to this position, before providing more detail. The final two sentences speak broad implications and importance.

---

Foreign-aid projects typically have local effects, so they need to be placed close to the poor if they are to reduce poverty. I show that, conditional on local population levels, World Bank (WB) project aid targets richer parts of countries. This relationship holds over time and across world regions. I test five donor-side explanations for pro-rich targeting using a pre-registered conjoint experiment on WB Task Team Leaders (TTLs). TTLs perceive aid-receiving governments as most interested in targeting aid politically and controlling implementation. They also believe that aid works better in poorer or more remote areas, but that implementation in these areas is uniquely difficult. These results speak to debates in distributive politics, international bargaining over aid, and principal-agent issues in international organizations. The results also suggest that tweaks to WB incentive structures to make ease of project implementation less important may encourage aid to flow to poorer parts of countries.

---

The journal *Nature* provides a guide for constructing an abstract. They recommend a structure that results in an abstract of six parts, that add up to around 200 words.

- 1) A basic introductory sentence that is comprehensible to a wide audience.
- 2) A more detailed sentence about background that is relevant to likely readers.
- 3) A sentence that states the general problem.
- 4) Sentences that summarize and then explain the main results.
- 5) A sentence about general context.
- 6) And finally, a sentence about the broader perspective.

### 5.3.3 Introduction

An introduction needs to be self-contained and convey everything that a reader needs to know. It is important to recognize that we are not writing a mystery story. Instead, we want to give-away the most important points in the introduction. For a six-page paper, an introduction may be two or three paragraphs

of main content. Hayot (2014, p. 90) describes the goal of an introduction is to engage the reader, locate them in some discipline and background, and then tell them what happens in the rest of the paper. It is completely reader-focused.

The introduction should set the scene and give the reader some background. For instance, we typically start a little broader. This provides some context to the paper. We then describe how the paper fits into that context, and give some high-level results, especially focused on the one key result that is the main part of the story. We provide more detail here than we provided in the abstract, but not the full extent. And the final bit of main content is to broadly discuss next steps. Finally, we finish the introduction with an additional short final paragraph that highlights the structure of the paper.

As an example (with made-up details):

---

The UK Conservative Party has always done well in rural electorates. And the 2016 Brexit vote was no different with a significant difference in support between rural and urban areas. But even by the standard of rural support for conservative issues, support for ‘Vote Leave’ was unusually strong with ‘Vote Leave’ being most heavily supported in the East Midlands and the East of England, while the strongest support for ‘Remain’ was in Greater London.

In this paper we look at why the performance of ‘Vote Leave’ in the 2016 Brexit referendum was so correlated with rurality. We construct a model in which support for ‘Vote Leave’ at a voting area level, is explained by the number of farms in the area, the average internet connectivity, and the median age. We find that as the median age of an area increases, the likelihood that an area supported ‘Vote Leave’ decreases by 14 percentage points. Future work could look at the effect of having a Conservative MP which would allow a more nuanced understanding of these effects.

The remainder of this paper is structured as follows: Section 2 discusses the data, Section 3 discusses the model, Section 4 presents the results, and finally Section 5 discusses our findings and some weaknesses.

---

The introduction needs to be self-contained and tell your reader everything

that they need to know. A reader should be able to only read the introduction and have an accurate picture of all the major aspects that they would if they were to read the whole paper. It would be rare to include graphs or tables in the introduction. An introduction always closes with the structure of the paper. For instance (and this is just a rough guide) an introduction for a 10-page paper, should probably be about 3 or 4 paragraphs, or 10 per cent, but it depends on specifics.

### 5.3.4 Data

Robert Caro, Lyndon B. Johnson's biographer, describes the importance of conveying 'a sense of place' when writing biography (Caro, 2019, p. 141). This he defines as 'the physical setting in which a book's action is occurring: to see it clearly enough, in sufficient detail, so that he feels as if he himself were present while the action is occurring.' He provides the following example:

---

When Rebekah walked out the front door of that little house, there was nothing—a roadrunner streaking behind some rocks with something long and wet dangling from his beak, perhaps, or a rabbit disappearing around a bush so fast that all she really saw was the flash of a white tail—but otherwise nothing. There was no movement except for the ripple of the leaves in the scattered trees, no sound except for the constant whisper of the wind... If Rebekah climbed, almost in desperation, the hill in the back of the house, what she saw from its crest was more hills, an endless vista of hills, hills on which there was visible not a single house... hills on which nothing moved, empty hills with, above them, empty sky; a hawk circling silently overhead was an event. But most of all, there was nothing human, no one to talk to.

Caro (2019, p. 146)

---

How thoroughly we can imagine the circumstances of Rebekah Baines Johnson (Lyndon B. Johnson's mother). We need to provide our reader with the same sense of place for our dataset. When writing our papers, we need to achieve that same sense of place, for our data, as Caro is able to provide for the Hill county. We do this by being as explicit as possible about showing our dataset. We typically have a whole section about it and this is designed to show the reader, as closely as possible, the actual data that underpin our story.

When writing the data section, we are beginning our answer to the critical question about our claims, which is, how is it possible to know this? (McPhee, 2017, p. 78). The preeminent example of a data section is provided by Doll and Hill (1950), who are interested in the effect smoking between control and treatment groups. They begin by clearly describing their dataset. They then use tables to display relevant cross-tabs. And use graphs to contrast their groups.

In the data section we need to thoroughly discuss the variables in the dataset that we are using. If there are other datasets that could have been used, but were not, then these should be mentioned and our choices justified. If variables were constructed or combined, then this process and motivation should be explained.

To get a sense of the data, it is important that the reader is able to understand what the data that underpin the results look like. This means that we should graph the actual data that are used in our analysis, or as close to them as possible. And we should also include tables of summary statistics. If the dataset was created from some other source, then it can also help to include an example of that original source. For instance, if the dataset was created from survey responses then the survey form should be included, potentially in an appendix.

The data section will also have figures and tables. Here some judgment is required. While it is important that the reader has the opportunity to understand the details, it may be that some are better placed in an appendix. Figure and tables are a critical aspect of convincing people of a story. In a graph we can show the data and then let the reader decide for themselves. And using a table, we can more easily summarize our dataset. At the very least, every variable needs to be shown in a graph and summarized in a table. Figures and tables should be numbered and then cross-referenced in the text, for instance, “Figure 1 shows...”, “Table 1 describes...”. For every graph and table there should be extensive accompanying text that describes their main aspects, and adds additional detail.

We discuss the components of graphs and tables, including titles and labels, in Chapter 6. But here we will discuss captions, as they are between text and the graph or table. Captions need to be informative and self-contained. As Cleveland (1994, p. 57) says, the ‘interplay between graph, caption, and text is a delicate one’, however the reader should be able to read only the caption and understand what the graph or table shows. A caption that is two of three lines long would be inappropriate. And all aspects of the graph or table should be explained. For instance, consider Figures 5.1 and 5.2 from Bowley (1901, p. 151), which are both exceptionally clear, and self-contained.

The choice between a table and a graph comes down to how much information is to be conveyed. In general, if there is specific information that should be

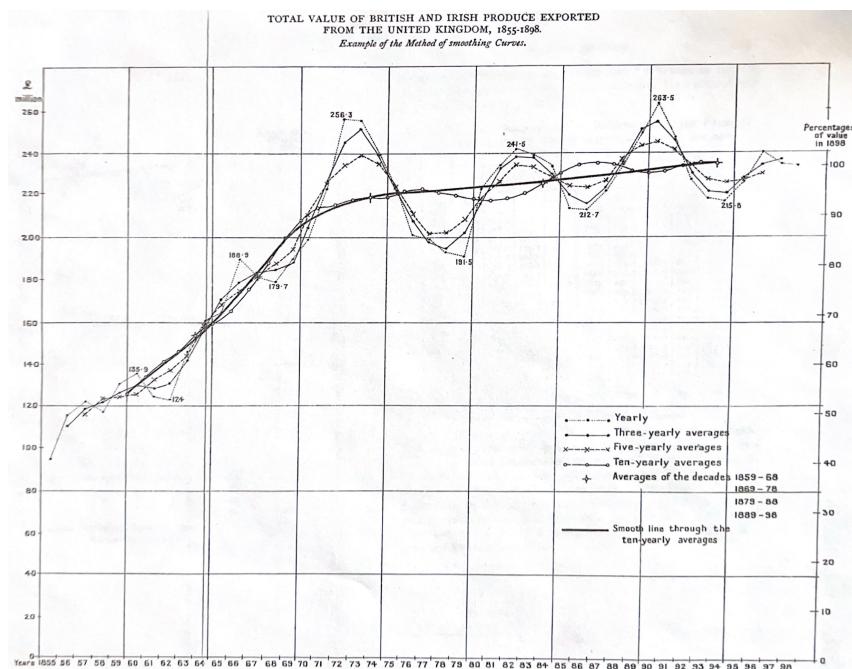


FIGURE 5.1: Example of a well-captioned figure

TOTAL DECLARED REAL VALUE OF BRITISH AND IRISH PRODUCE  
EXPORTED FROM THE UNITED KINGDOM.  $1 = \text{£}1,000,000$ .

		Averages.				Averages.		
		Three Yearly.	Five Yearly.	Ten Yearly.		Three Yearly.	Five Yearly.	Ten Yearly.
		1855	95.7	...	1878	192.8	210.9	218.0
1856	115.8	...	...	...	1879	191.5	194.4	201.4
1857	122.0	111.2	...	...	1880	223.1	202.5	201.3
1858	116.6	118.1	...	...	1881	234.0	216.2	208.2
1859	130.4	123.0	116.1	...	1882	241.5	232.9	216.7
1860	135.9	127.6	124.1	...	1883	239.8	238.4	226.0
1861	125.1	130.5	126.0	...	1884	233.0	238.1	234.3
1862	124.0	128.3	126.4	...	1885	213.1	228.6	232.3
1863	146.5	131.9	132.4	...	1886	212.7	219.6	228.0
1864	160.4	143.7	138.4	127.2	1887	221.9	215.6	224.1
1865	165.8	157.6	144.4	134.3	1888	234.5	223.0	223.0
1866	188.9	171.7	157.2	141.6	1889	248.9	235.1	226.2
1867	181.0	178.6	168.7	147.5	1890	263.5	249.0	236.3
1868	179.7	183.2	175.1	153.8	1891	247.2	253.2	243.2
1869	190.0	183.6	181.0	159.8	1892	227.1	245.9	244.2
1870	199.6	189.8	187.8	165.9	1893	218.1	230.8	240.9
1871	223.1	204.2	194.6	175.7	1894	215.8	220.3	234.3
1872	256.3	226.3	209.7	188.9	1895	225.9	219.9	226.8
1873	255.2	244.9	224.8	200.0	1896	240.1	227.3	225.4
1874	239.6	250.4	234.7	207.9	1897	234.3	233.4	226.8
1875	223.5	239.4	239.6	213.7	1898	233.4	235.9	229.8
1876	200.5	221.0	235.1	214.9	1899	255.4*	241.0	237.8
1877	198.9	207.7	223.7	216.7				236.1

\* Not including the newly reckoned value of ships exported.

FIGURE 5.2: Example of a well-captioned table

considered, such as a summary statistic, then a table is a good option, while if we are interested in the reader making comparisons and understanding trends then a graph is a good option (Gelman et al., 2002).

Finally, if there is relevant literature then we would discuss it throughout the paper as appropriate. For instance, when there is literature relevant to the data then it should be discussed in this section, literature relevant to the model, results, or discussion should be mentioned as appropriate in those sections. It is rarely necessary to have a separate literature review section.

### 5.3.5 Model

We will often build a statistical model that we will use to explore the data, and we often have a specific section about this. At a minimum it is important to clearly specify equation/s that describe the model being used, and explain their components with plain language and cross-references.

The model section typically begins with the model being written out, explained, and justified. Depending on the expected reader, some background may be needed. After specifying the model with appropriate mathematical notation and cross-referencing it, the components of the model are then typically defined and explained. It is especially important to define each aspect of the notation. This helps convince the reader that the model was well-chosen and enhances the credibility of the paper. The model's variables should correspond to those that were discussed in the data section, making a clear link between the two sections.

There should be some discussion of how features enter the model and why. For instance, some examples could include, why use ages rather than age-groups, why does state/province have a levels effect, and why is gender a categorical variable. In general, we are trying to convey a sense that this is the model for the situation. We want the reader to understand how the aspects that were discussed in the data section assert themselves in the modelling decisions that were made.

The model section should close with some discussion of the assumptions that underpin the model, and a brief discussion of alternative models, or variants, and strengths and weaknesses made clear. It should be clear in the reader's mind why it was this model that was chosen.

At some point in this section, it is usually appropriate to specify the software that was used to run the model, and to provide some evidence of thought about the circumstances in which the model may not be appropriate. The later point would typically be expanded on in the discussion. And there should be evidence of model validation and checking, model convergence, and/or diagnostic issues. Again, there is a balance needed here, and some of this content may be more appropriate placed in appendices.

When technical terms are used, they should be briefly explained in plain language for readers who might not be familiar with it. For instance, Alexander (2019b) integrates an explanation of the Gini coefficient that brings the reader along.

---

To look at the concentration of baby names, let's calculate the Gini coefficient for each country, sex and year. The Gini coefficient measures dispersion or inequality among values of a frequency distribution. It can take any value between 0 and 1. In the case of income distributions, a Gini coefficient of 1 would mean one person has all the income. In this case, a Gini coefficient of 1 would mean that all babies have the same name. In contrast, a Gini coefficient of 0 would mean names are evenly distributed across all babies.

---

### 5.3.6 Results

Two excellent examples of results sections provided by Kharecha and Hansen (2013) and Kiang et al. (2021). In the results section, we want to communicate the outcomes of the model in a clear way and without too much in the way of discussion of implications. The results section likely requires summary statistics, tables, and graphs. Each of those aspects should be cross-referenced and have text associated with them that details what is seen in them. This section should strictly relay results; that is, we are interested in what the results are, rather than what they mean.

This section would also typically include table/s of coefficient estimates based on the modelling that we used to further explore the data. Various features of the estimates should be discussed, and differences between the models explained. It may be that different subsets of the data are considered separately. Again, all graphs and tables need to have plain language text accompany them. A rough guide is that the amount of text should be at least equal to the amount of space taken up by the tables and graphs. For instance, if a full page is used to display a table of coefficient estimates, then that should be cross-referenced and accompanied by at least a full page of text about that table.

### 5.3.7 Discussion

A discussion section may be the final section of a paper and would typically have four or five sub-sections.

The discussion section would typically begin with a sub-section that comprises a one- or two-paragraph summary of what was done in the paper. This would be followed by two or three sub-sections that are devoted to the key things that we learn about the world from this paper. For instance, there are typically a few implications that come from the modelling results. These few sub-sections are the main opportunity to justify or detail the implications of the story being told in the paper. Typically, these sub-sections do not see newly introduced graphs or tables, but are instead focused on what we learn from those that were introduced in earlier sections. It may be that some of the results are discussed in relation to what others have found, and differences could be attempted to be reconciled here.

Following these sub-sections of what we learn about the world, we would typically have a sub-section focused on some of the weaknesses of what was done. This could concern aspects such as the data that were used, the approach, and the model. And the final sub-section is typically a few paragraphs that specify what is left to learn, and how future work could proceed.

In general, we would expect this section to take at least twenty-five per cent of the total paper. For instance, in an eight page paper, we would expect at least two pages of discussion.

### 5.3.8 Brevity, typos, and grammar

Brevity is important. Partly this is because we write for the reader, and the reader has other priorities. But it is also because as the writer it focuses us to consider what our most important points are, how we can best support them, and where our arguments are weakest. Jean Chrétien, the former Canadian Prime Minister, describes how ‘[t]o allow me to get to the heart of an issue quickly, I asked the officials to summarize their documents in two or three pages and attach the rest of the materials as background information. I soon discovered that this was a problem only for those who didn’t really know what they were talking about.’ ([Chrétien, 2007](#), p. 105).

This experience is not unique to Canada. For instance, Oliver Letwin, the former British Conservative Cabinet member, describes there as being ‘a huge amount of terrible guff, at huge, colossal, humongous length coming from some departments’ and how he asked ‘for them to be one quarter of the length’ ([Hughes and Rutter, 2016](#)). He found that the departments were able to accommodate this request without losing anything important.

This experience is also not new. For instance, Churchill asked for brevity during the Second World War, saying ‘the discipline of setting out the real points concisely will prove an aid to clearer thinking’. And the letter from Szilard and Einstein to FDR that was the catalyst for the Manhattan Project was only two pages.

This experience is also not unique to academia. For instance, one of the foundations of Amazon, which is one of the world's largest companies, is clear writing. Specifically, instead of PowerPoint presentations, Jeff Bezos asked for '[w]ell structured, narrative text... [which] forces better thought and better understanding of what's more important than what, and how things are related.'

Zinsser (1976) goes further and describes 'the secret of good writing' being 'to strip every sentence to its cleanest components.' Every sentence should be simplified to its essence. And every word that does not contribute should be removed.

Typos and other grammatical mistakes affect the credibility of claims. If the reader cannot trust us to use a spell-checker, then why should they trust us to use logistic regression? Microsoft Word and Google Docs are useful here for their spell-checkers: copy/paste from R Markdown, look for the red and green lines, and fix them in R Markdown.

We are not worried about the n-th degree of grammatical content. Instead, we are interested in grammar and sentence structure that occurs in conversational language use (King, 2000, p. 118). The way to develop that comfort is by reading a lot, and asking others to read your work also.

Unnecessary words, typos, and grammatical issues should be removed from papers with a fanatical zeal.

### 5.3.9 Rules

A variety of authors have established rules for writing, including famously, Orwell (1946), which were reimagined by The Economist (2013). And Fiske and Kuriwaki (2021) have a list of rules for scientific papers. A further reimagining, focused on telling stories with data, could be:

- Focus on the reader and their needs. Everything else is comment.
- Establish a logical structure and rely on that structure to tell the story.
- Write a first draft as quickly as possible.
- Re-write that extensively and without favor.
- Aim to be concise and direct. Remove as many words as possible.
- Using words precisely. Stock-markets rise or fall, not improve or worsen.
- Use short sentence where possible.
- Avoid jargon.
- Write as though your work will be on the front page of a newspaper. Because it could be.

## 5.4 Exercises and tutorial

### 5.4.1 Exercises

1. According to King (2006), what is the key task of subheadings (pick one)?
  - a. Enable a reader who randomly falls asleep but keeps turning pages to know where they are.
  - b. Be broad and sweeping so that a reader is impressed by the importance of the paper.
  - c. Use acronyms to integrate the paper into the literature.
2. According to King (2006), what is the maximum length of an abstract (pick one)?
  - a. Two hundred words.
  - b. Two hundred and fifty words.
  - c. One hundred words.
  - d. One hundred and fifty words.
3. According to King (2006), in a paper, raw computer output should be (pick one)?
  - a. Commented out.
  - b. Not included.
  - c. Included.
4. According to King (2006), if our standard error was 0.05 then which of the following specificity for a coefficient would be silly (select all that apply)?
  - a. 2.7182818
  - b. 2.718282
  - c. 2.72
  - d. 2.7
  - e. 2.7183
  - f. 2.718
  - g. 3
  - h. 2.71828
5. When should we try not to use the ‘delete’ key (pick one)?
  - a. While writing the first draft.
  - b. While writing the second draft.
  - c. While writing the third draft.
  - d. The ‘delete’ key should always be used.
6. How long should a first draft take to write of a five-to-ten-page paper (pick one)?
  - a. One hour
  - b. One day
  - c. One week

- d. One month
7. What is a key aspect of the re-drafting process (select all that apply)?
- Going through it with a red pen to remove unneeded words.
  - Printing the paper and reading a physical copy.
  - Cutting and pasting to enhance flow.
  - Reading it aloud.
  - Exchanging it with others.
8. What are three features of a good research question (write a paragraph or two)?
9. What are some of the challenges of being ‘data-first’ (write a paragraph or two)?
10. What are some of the challenges of being ‘question-first’ (write a paragraph or two)?
11. What is a counterfactual (pick one)?
- If-then statements in which the if does not happen.
  - If-then statements in which the if happens.
  - Statements that are either true or false.
  - Statements that are neither true or false.
12. Which of the following is the best title (pick one)?
- “Problem Set 1”
  - “Unemployment”
  - “Examining England’s Unemployment (2010-2020)”
  - “England’s Unemployment Increased between 2010 and 2020”
13. Which of the following is the best title (pick one)?
- “Problem Set 2”
  - “Standard errors”
  - “On standard errors with small samples”
14. Which word/s can be removed from the following sentence without affecting its meaning (select all that apply)? ‘Like many parents, when our children were born, one of the first things that my wife and I did regularly was read stories to them.’
- first
  - regularly
  - stories
15. Please write a new title for either [Barron et al. \(2018\)](#) or [Fourcade and Healy \(2017\)](#).
16. Please write a new title for the first article from the list of articles from *The New Yorker* that you read.
17. Please write a new title for the other article from the list of articles from *The New Yorker* that you read.
18. Please write a new four-sentence abstract for [Chambliss \(1989\)](#)
19. Please write a new four-sentence abstract for [Doll and Hill \(1950\)](#) or [Student \(1908\)](#) or [Kharecha and Hansen \(2013\)](#).

20. Please write an abstract for the first article from the list of ‘miscellaneous’ articles that you read.
  21. Please write an abstract for the other article from the list of ‘miscellaneous’ articles that you read.
  22. Using only the 1000-most popular words in the English language – <https://xkcd.com/simplewriter/> – re-write the following so that it retains its original meaning:
- 

When using data, we try to tell a convincing story. It may be as exciting as predicting elections, as banal as increasing internet advertising click rates, as serious as finding the cause of a disease, or as fun as forecasting basketball games. In any case the key elements are the same.

---

#### 5.4.2 Tutorial

Caro (2019, p. xii) writes at least one thousand words almost every day. In this tutorial we will write every day for a week. Begin by picking seven of the well-written papers specified above. Each day complete the following tasks:

- Transcribe, by writing each word yourself, the entire introduction.
- (This idea comes from McPhee (2017, p. 186).) Re-write the introduction so that it is five lines (or 10 per cent, whichever is less) shorter.
- Transcribe, by writing each word yourself, the abstract.
- Re-write a new, four-sentence, abstract for the paper.
- (This idea comes from Chelsea Parlett-Pelleriti.) Write a second version of your new abstract using only the one-thousand most popular words in the English language: <https://xkcd.com/simplewriter/>.
- Detail three points about the way the paper is written that you like
- Detail one point about the way the paper is written that you do not like.

Submit all seven papers.



# 6

---

## Static communication

---

### Required material

- Read *R for Data Science*, Chapter 28 ‘Graphics for communication’, ([Wickham and Grolemund, 2017](#)).
- Read *Data Visualization: A Practical Introduction*, Chapters 3 ‘Make a plot’, 4 ‘Show the right numbers’, and 5 ‘Graph tables, add labels, make notes’, ([Healy, 2018](#)).
- Read *Testing Statistical Charts: What Makes a Good Graph?*, ([Vanderplas et al., 2020](#)).
- Read *Data Feminism*, Chapter 3 ‘On Rational, Scientific, Objective Viewpoints from Mythical, Imaginary, Impossible Standpoints’, ([D’Ignazio and Klein, 2020](#)).

### Key concepts and skills

- Knowing the importance of showing the reader the actual dataset, or as close as is possible.
- Using a variety of different graph options, including bar charts, scatterplots, line plots, and histograms.
- Knowing how to use tables to show part of a dataset, communicate summary statistics, and display regression results.
- Approaching maps as a type of a graph.
- Comfort with geocoding places.

### Key libraries

- `datasauRus` ([Locke and D’Agostino McGowan, 2018](#))
- `ggmap` ([Kahle and Wickham, 2013](#))
- `kableExtra` ([Zhu, 2020](#))
- `knitr` ([Xie, 2021](#))
- `maps`
- `modelsummary` ([Arel-Bundock, 2021a](#))
- `opendatatoronto` ([Gelfand, 2020](#))
- `patchwork` ([Pedersen, 2020](#))
- `tidyverse` ([Wickham et al., 2019a](#))
- `viridis` ([Garnier et al., 2021](#))
- `WDI` ([Arel-Bundock, 2021b](#))

### Key functions

- `ggmap::get_googlemap()`
- `ggmap::get_stamenmap()`
- `ggmap::ggmap()`
- `ggplot2::coord_map()`
- `ggplot2::facet_wrap()`
- `ggplot2::geom_abline()`
- `ggplot2::geom_bar()`
- `ggplot2::geom_boxplot()`
- `ggplot2::geom_dotplot()`
- `ggplot2::geom_freqpoly()`
- `ggplot2::geom_histogram()`
- `ggplot2::geom_jitter()`
- `ggplot2::geom_line()`
- `ggplot2::geom_path()`
- `ggplot2::geom_point()`
- `ggplot2::geom_polygon()`
- `ggplot2::geom_smooth()`
- `ggplot2::geom_step()`
- `ggplot2::ggplot()`
- `ggplot2::ggsave()`
- `ggplot2::labeller()`
- `ggplot2::labs()`
- `ggplot2::map_data()`
- `ggplot2::scale_color_brewer()`
- `ggplot2::scale_colour_viridis_d()`
- `ggplot2::scale_fill_brewer()`
- `ggplot2::scale_fill_viridis()`
- `ggplot2::stat_qq()`
- `ggplot2::stat_qq_line()`
- `ggplot2::theme()`
- `ggplot2::theme_bw()`
- `ggplot2::theme_classic()`
- `ggplot2::theme_linedraw()`
- `ggplot2::theme_minimal()`
- `kableExtra::add_header_above()`
- `knitr::kable()`
- `lm()`
- `maps::map()`
- `modelsummary::datasummary()`
- `modelsummary::datasummary_balance()`
- `modelsummary::datasummary_correlation()`
- `modelsummary::datasummary_skim()`
- `modelsummary::modelsummary()`

- `WDI::WDI()`
  - `WDI::WDIsearch()`
- 

## 6.1 Introduction

When telling stories with data, we would like the data to do much of the work of convincing our reader. The paper is the medium, and the data are the message. To that end, we want to try to show our reader the data that allowed us to come to our understanding of the story. We use graphs, tables, and maps to help achieve this.

The critical task is to show the actual data that underpin our analysis, or as close to it as we can. For instance, if our dataset consists of 2,500 responses to a survey, then at some point in our paper we would expect a graph that contains 2,500 points. To do this we build graphs using `ggplot2` (Wickham, 2016). We will go through a variety of different options here including bar charts, scatterplots, line plots, and histograms.

In contrast to the role of graphs, which is to show the actual data, or as close to it as possible, the role of tables is typically to show an extract of the dataset or convey various summary statistics. We will build tables using `knitr` (Xie, 2021) and `kableExtra` (Zhu, 2020) initially, and then `modelsummary` (Arel-Bundock, 2021a).

Finally, we cover maps as a variant of graphs that are used to show a particular type of data. We will build static maps using `ggmap` (Kahle and Wickham, 2013), having obtained the geocoded data that we need using `tidygeocoder` (Cambon and Belanger, 2021).

---

## 6.2 Graphs

Graphs are a critical aspect of compelling stories told with data.

---

Graphs allow us to explore data to see overall patterns and to see detailed behavior; no other approach can compete in revealing the structure of data so thoroughly. Graphs allow us to view complex mathematical models fitted to data, and they allow us to assess the validity of such models.

Cleveland (1994, p. 5)

In a way, the graphing of data is an information coding process where we create a glyph, or purposeful mark, that we mean to convey information to our audience. The audience must decode our glyph. The success of our graph turns on how much information is lost in this process. It is the decoding that is the critical aspect (Cleveland, 1994, p. 221), which means that we are creating graphs for the audience. If nothing else is possible, the most important feature is to convey as much of the actual data as possible.

To see why this is important we begin by using the dataset ‘datasaurus\_dozen’ from `datasauRus` (Locke and D’Agostino McGowan, 2018). After installing and loading the necessary packages, we can take a quick look at the dataset.

```
install.packages('datasauRus')
```

```
library(tidyverse)
library(datasauRus)

head(datasaurus_dozen)
#> # A tibble: 6 x 3
#> dataset x y
#> <chr> <dbl> <dbl>
#> 1 dino 55.4 97.2
#> 2 dino 51.5 96.0
#> 3 dino 46.2 94.5
#> 4 dino 42.8 91.4
#> 5 dino 40.8 88.3
#> 6 dino 38.7 84.9
datasaurus_dozen |>
 count(dataset)
#> # A tibble: 13 x 2
#> dataset n
#> <chr> <int>
#> 1 away 142
#> 2 bullseye 142
#> 3 circle 142
#> 4 dino 142
#> 5 dots 142
#> 6 h_lines 142
#> 7 high_lines 142
```

```
#> 8 slant_down 142
#> 9 slant_up 142
#> 10 star 142
#> 11 v_lines 142
#> 12 wide_lines 142
#> 13 x_shape 142
```

We can see that the dataset consists of values for ‘x’ and ‘y’, which should be plotted on the x-axis and y-axis, respectively. We can further see that there are thirteen different values in the variable ‘dataset’ including: “dino”, “star”, “away”, and “bullseye”. We will focus on those four and generate summary statistics for each (Table 6.1).

```
From Julia Silge:
https://juliasilge.com/blog/datasaurus-multiclass/
datasaurus_dozen |>
 filter(dataset %in% c("dino", "star", "away", "bullseye")) |>
 group_by(dataset) |>
 summarise(across(c(x, y),
 list(mean = mean,
 sd = sd)),
 x_y_cor = cor(x, y)) |>
 knitr::kable(
 caption =
 "Mean and standard deviation for four 'datasaurus' datasets",
 col.names = c("Dataset",
 "x mean",
 "x sd",
 "y mean",
 "y sd",
 "correlation"),
 digits = 1,
 booktabs = TRUE,
 linesep = ""
)
```

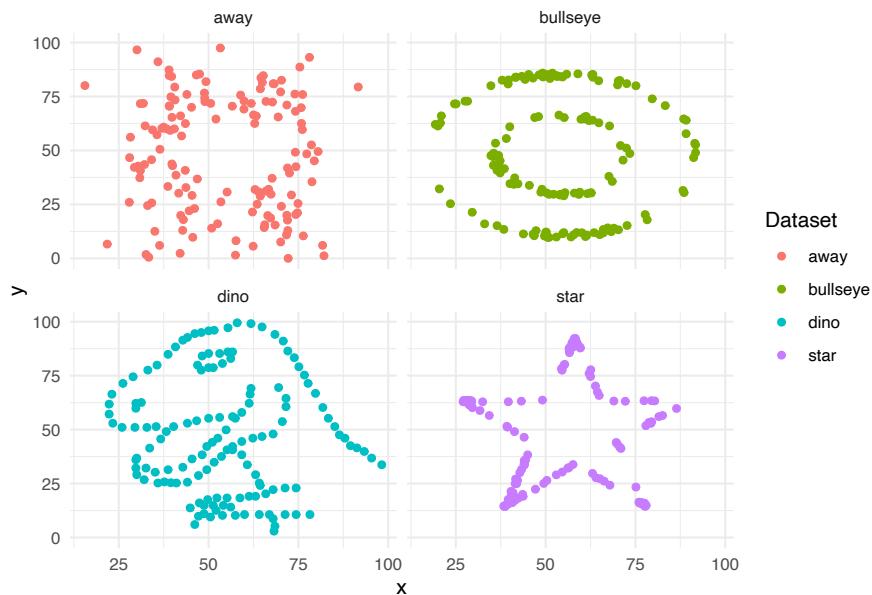
Despite the similarities of the summary statistics, it turns out the different ‘datasets’ are actually very different beasts when we graph the actual data (Figure 6.1).

```
datasaurus_dozen |>
 filter(dataset %in% c("dino", "star", "away", "bullseye")) |>
 ggplot(aes(x=x, y=y, colour=dataset)) +
```

**TABLE 6.1:** Mean and standard deviation for four 'datasaurus' datasets

Dataset	x mean	x sd	y mean	y sd	correlation
away	54.3	16.8	47.8	26.9	-0.1
bullseye	54.3	16.8	47.8	26.9	-0.1
dino	54.3	16.8	47.8	26.9	-0.1
star	54.3	16.8	47.8	26.9	-0.1

```
geom_point() +
theme_minimal() +
facet_wrap(vars(dataset), nrow = 2, ncol = 2) +
labs(colour = "Dataset")
```

**FIGURE 6.1:** Graph of four 'datasaurus' datasets

This is a variant of the famous 'Anscombe's Quartet'. The key takeaway is that it is important to plot the actual data and not rely on summary statistics. The 'anscombe' dataset is built into R.

```
head(anscombe)
#> x1 x2 x3 x4 y1 y2 y3 y4
```

```
#> 1 10 10 10 8 8.04 9.14 7.46 6.58
#> 2 8 8 8 8 6.95 8.14 6.77 5.76
#> 3 13 13 13 8 7.58 8.74 12.74 7.71
#> 4 9 9 9 8 8.81 8.77 7.11 8.84
#> 5 11 11 11 8 8.33 9.26 7.81 8.47
#> 6 14 14 14 8 9.96 8.10 8.84 7.04
```

It consists of six observations for four different datasets, again with x and y values for each observation. We need to manipulate this dataset with `pivot_longer()` to get it into a ‘tidy format’.

```
From Nick Tierney:
https://www.njtierney.com/post/2020/06/01/tidy-anscombe/
Code from pivot_longer() vignette.
tidy_anscombe <-
 anscombe |>
 pivot_longer(everything(),
 names_to = c(".value", "set"),
 names_pattern = "(.)(.)")
)
```

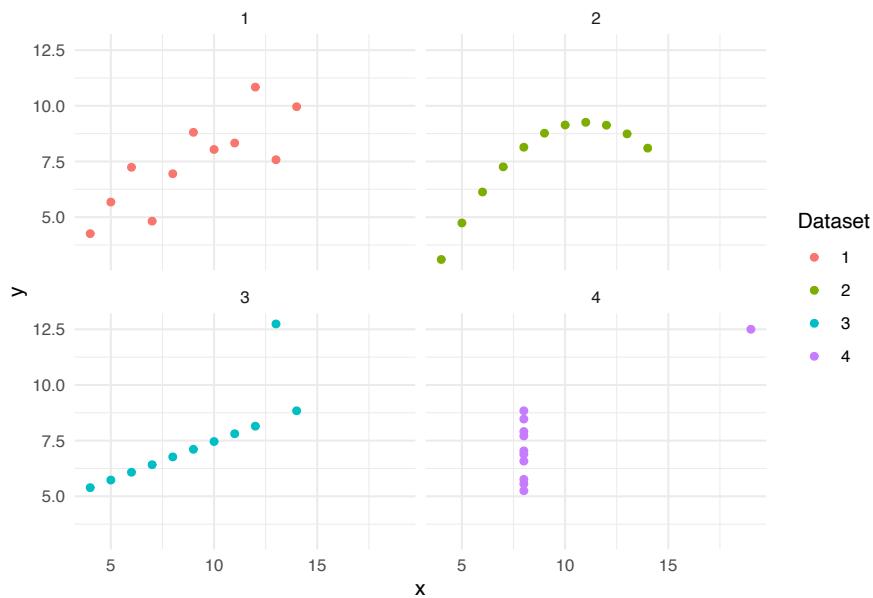
We can again first create some summary statistics (Table 6.2) and then graph the data (Figure 6.2). And we again see the importance of graphing the actual data, rather than relying on summary statistics.

```
tidy_anscombe |>
 group_by(set) |>
 summarise(across(c(x, y),
 list(mean = mean, sd = sd)),
 x_y_cor = cor(x, y)) |>
 knitr::kable(
 caption = "Mean and standard deviation for Anscombe",
 col.names = c("Dataset",
 "x mean",
 "x sd",
 "y mean",
 "y sd",
 "correlation"),
 digits = 1,
 booktabs = TRUE,
 linesep = "")
)
```

**TABLE 6.2:** Mean and standard deviation for Anscombe

Dataset	x mean	x sd	y mean	y sd	correlation
1	9	3.3	7.5	2	0.8
2	9	3.3	7.5	2	0.8
3	9	3.3	7.5	2	0.8
4	9	3.3	7.5	2	0.8

```
tidy_anscombe |>
 ggplot(aes(x = x, y = y, colour = set)) +
 geom_point() +
 theme_minimal() +
 facet_wrap(vars(set), nrow = 2, ncol = 2) +
 labs(colour = "Dataset")
```

**FIGURE 6.2:** Recreation of Anscombe's Quartet

### 6.2.1 Bar charts

We typically use a bar chart when we have a categorical variable that we want to focus on. We saw an example of this in Chapter 2 where we constructed

a graph of the number of occupied beds. The geom that we primarily use is `geom_bar()`, but there are many variants to cater for specific situations.

We will use a dataset from the 1997-2001 British Election Panel Study that was put together by [Fox and Andersen \(2006\)](#).

```
Vincent Arel-Bundock provides access to this dataset.
beps <-
 read_csv(
 file =
 "https://vincentarelbundock.github.io/Rdatasets/csv/carData/BEPS.csv"
)

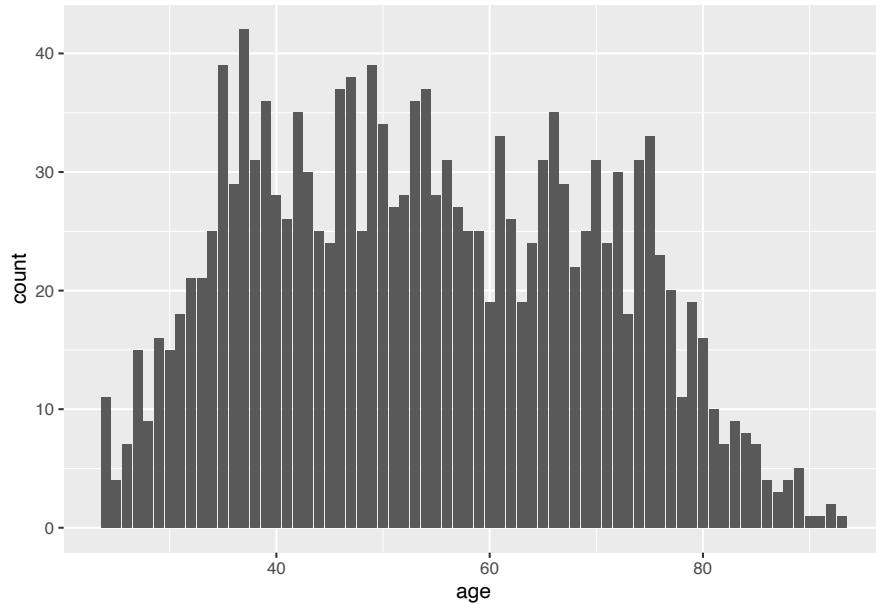
head(beps)
#> # A tibble: 6 x 11
#> ...1 vote age economic.cond.n~ economic.cond.h~ Blair
#> <dbl> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 1 Liber~ 43 3 3 4
#> 2 2 Labour 36 4 4 4
#> 3 3 Labour 35 4 4 5
#> 4 4 Labour 24 4 2 2
#> 5 5 Labour 41 2 2 1
#> 6 6 Labour 47 3 4 4
#> # ... with 5 more variables: Hague <dbl>, Kennedy <dbl>,
#> # Europe <dbl>, political.knowledge <dbl>, gender <chr>
```

The dataset consists of which party the person supports, along with various demographic, economic, and political variables. In particular, we have the age of the respondents. We could begin by making a graph of this age distribution using `geom_bar()` (Figure 6.3).

```
beps |>
 ggplot(mapping = aes(x = age)) +
 geom_bar()
```

By default, `geom_bar()` has created a count of the number of times each age appears in the dataset. It does this because the default ‘stat’ for `geom_bar()` is ‘count’. This saves us from having to create that statistic ourselves. But if we had already constructed a count (for instance, with `beps |> count(age)`), then we could also specify a column of values for the y-axis and then use `stat = "identity"`.

We may also like to consider different groupings of the data, for instance, looking at age-groups by which party the respondent supports (Figure 6.4).



**FIGURE 6.3:** Distribution of ages in the 1997-2001 British Election Panel Study

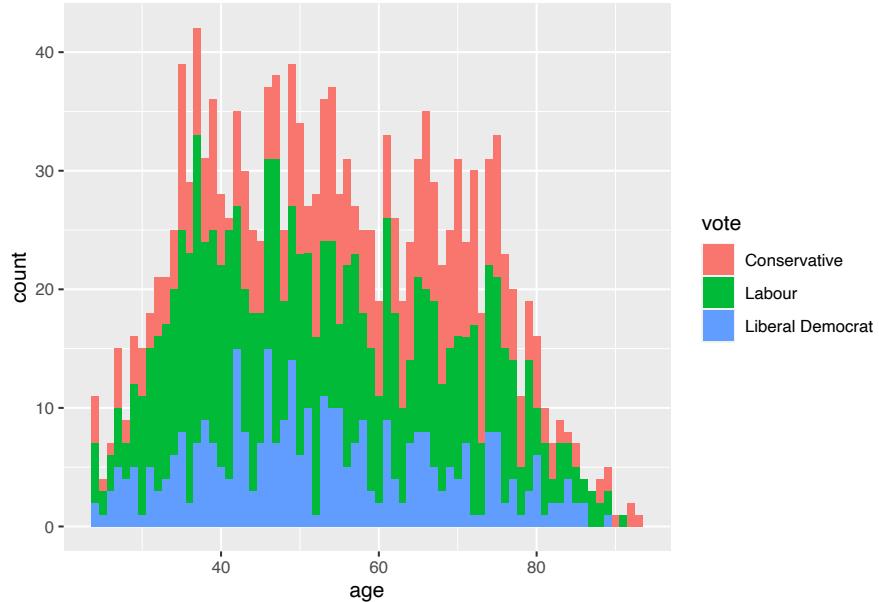
```
beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar()
```

The default is that these different groups are stacked, but they can be placed side-by-side with `position = "dodge"` (Figure 6.5).

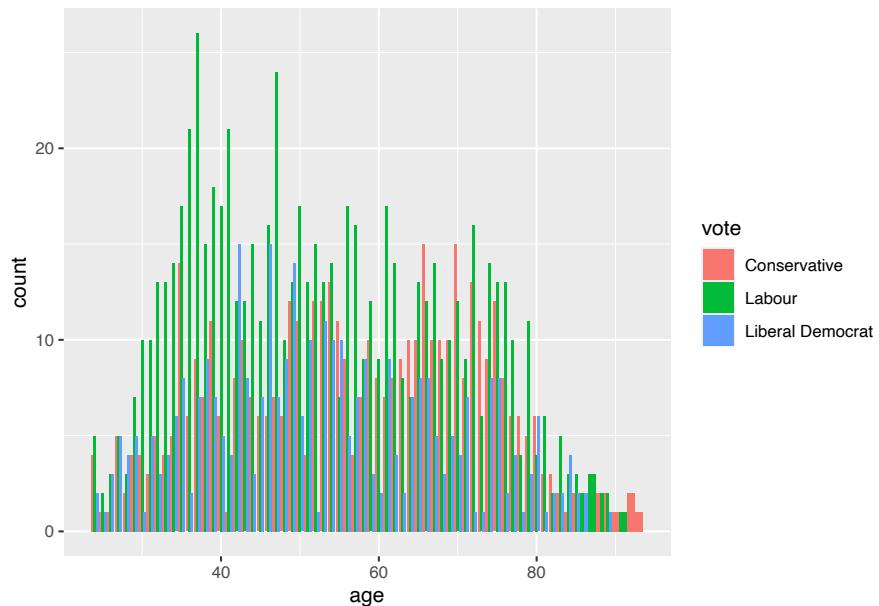
```
beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar(position = "dodge")
```

At this point, we may like to address the general look of the graph. There are various themes that are built into `ggplot2`. Some of these include `theme_bw()`, `theme_classic()`, `theme_dark()`, and `theme_minimal()`. A full list is available at the `ggplot2` cheatsheet<sup>1</sup>. We can use these themes by adding them as a layer (Figure 6.6). Here we can use `patchwork` (Pedersen, 2020) to bring together multiple graphs. To do this we assign the graph to a name, and then use ‘+’

<sup>1</sup><https://github.com/rstudio/cheatsheets/blob/main/data-visualization.pdf>



**FIGURE 6.4:** Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study



**FIGURE 6.5:** Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study

to signal which should be next to each other, ‘/’ to signal which would be on top, and brackets for precedence.

```
library(patchwork)

theme_bw <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar(position = "dodge") +
 theme_bw()

theme_classic <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar(position = "dodge") +
 theme_classic()

theme_dark <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar(position = "dodge") +
 theme_dark()

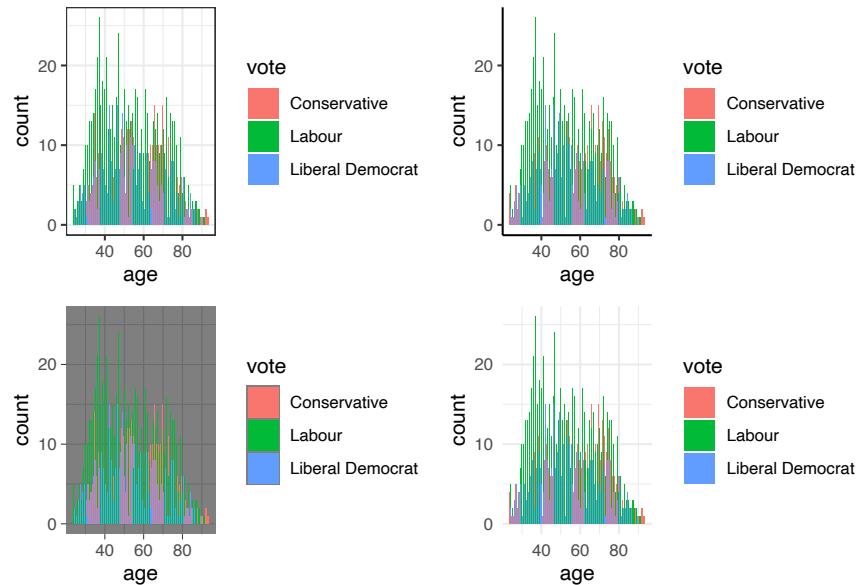
theme_minimal <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar(position = "dodge") +
 theme_minimal()

(theme_bw + theme_classic) / (theme_dark + theme_minimal)
```

We can install themes from other packages, including `ggthemes` (Arnold, 2021), and `hrbrthemes` (Rudis, 2020). And we can also build our own.

The default labels use `dby` `ggplot2` are from the name of the relevant variable, and it is often useful to add more detail. We could add a title and caption at this point. A caption can be useful to add information about the source of the dataset. A title can be useful when the graph is going to be considered outside of the context of our paper. But in the case of a graph that will be included in a paper, the need to cross-reference all graphs that are in a paper means that included a title within `labs()` is unnecessary (Figure 6.7).

```
beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
```

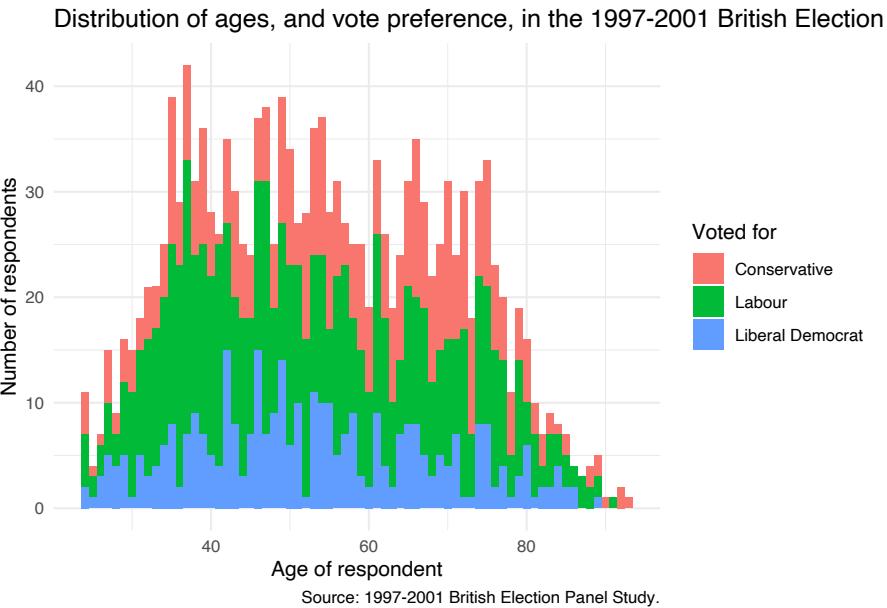


**FIGURE 6.6:** Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study, illustrating different themes

```
geom_bar() +
theme_minimal() +
labs(x = "Age of respondent",
y = "Number of respondents",
fill = "Voted for",
title = "Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study",
caption = "Source: 1997-2001 British Election Panel Study.")
```

We use facets to create ‘many little graphics that are variations of a single graphic’ (Wilkinson, 2005, p. 219). They are especially useful when we want to specifically compare across some variable, but have already used color. For instance, we may be interested to explain vote, by age and gender (Figure 6.8).

```
beps |>
ggplot(mapping = aes(x = age, fill = vote)) +
geom_bar() +
theme_minimal() +
labs(x = "Age of respondent",
```

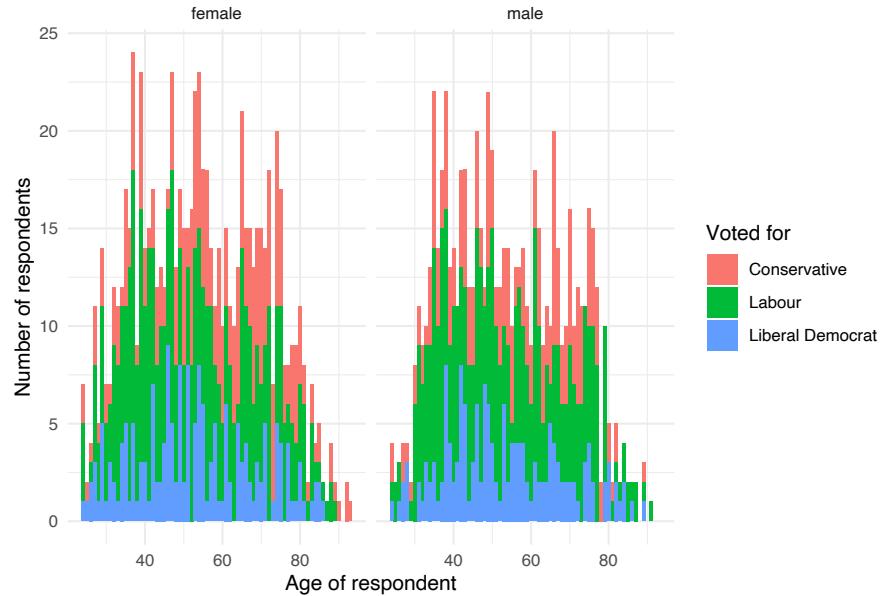


**FIGURE 6.7:** Distribution of ages, and vote preference, in the 1997-2001 British Election Panel Study

```
y = "Number of respondents",
fill = "Voted for") +
facet_wrap(vars(gender))
```

We could change `facet_wrap()` to wrap vertically instead of horizontally with `dir = "v"`. Alternatively, we could specify a number of rows, say `nrow = 2`, or a number of columns, say `ncol = 2`. Additionally, by default, both facets will have the same scales. We could enable both facets to have different scales with `scales = "free"`, or just the x-axis `scales = "free_x"`, or just the y-axis `scales = "free_y"` (Figure 6.9).

```
beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar() +
 theme_minimal() +
 labs(x = "Age of respondent",
 y = "Number of respondents",
 fill = "Voted for") +
 facet_wrap(vars(gender),
```



**FIGURE 6.8:** Distribution of age by gender, and vote preference, in the 1997-2001 British Election Panel Study

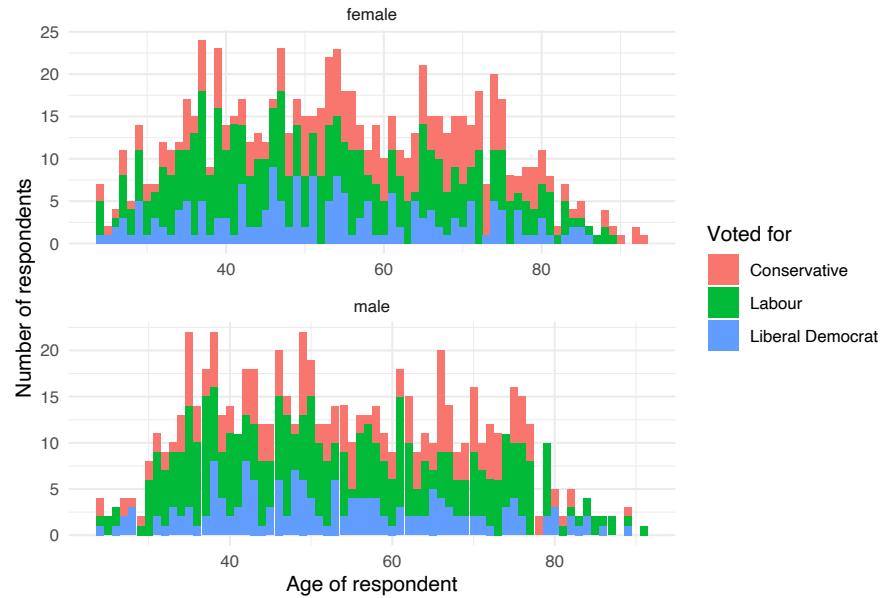
```
dir = "v",
scales = "free")
```

Finally, we can change the labels of the facets using `labeller()` (Figure 6.10).

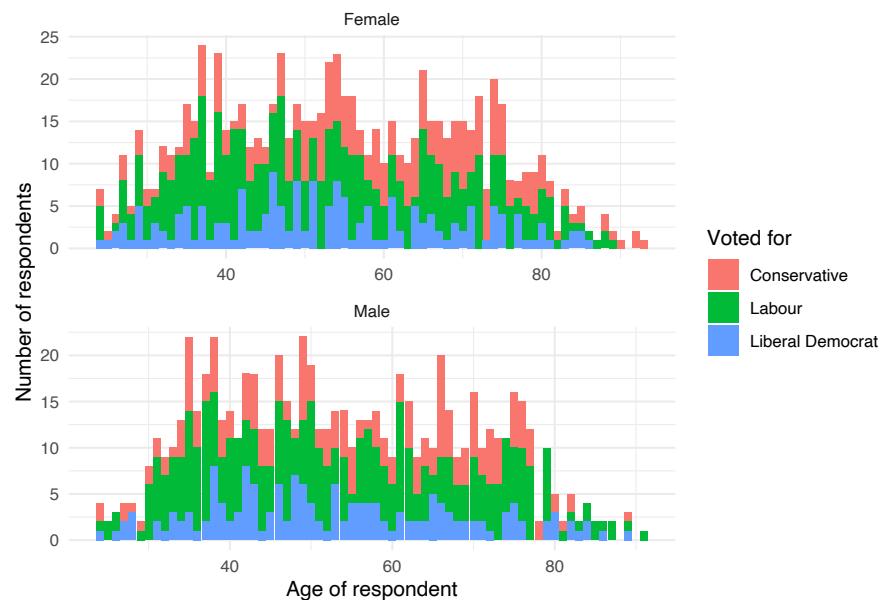
```
new_labels <- c(female = "Female", male = "Male")

beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar() +
 theme_minimal() +
 labs(x = "Age of respondent",
 y = "Number of respondents",
 fill = "Voted for") +
 facet_wrap(vars(gender),
 dir = "v",
 scales = "free",
 labeller = labeller(gender = new_labels))
```

There are a variety of different ways to change the colors, and many palettes



**FIGURE 6.9:** Distribution of age by gender, and vote preference, in the 1997-2001 British Election Panel Study



**FIGURE 6.10:** Distribution of age by gender, and vote preference, in the 1997-2001 British Election Panel Study

are available including from `RColorBrewer` (Neuwirth, 2014), which we specify with `scale_fill_brewer()`, and `viridis` (Garnier et al., 2021), which we specify with `scale_fill_viridis()` and is particularly focused on color-blind palettes (Figure 6.11).

```
library(viridis)
library(patchwork)

RColorBrewerBrBG <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar() +
 theme_minimal() +
 labs(x = "Age of respondent",
 y = "Number of respondents",
 fill = "Voted for") +
 scale_fill_brewer(palette = "Blues")

RColorBrewerSet2 <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar() +
 theme_minimal() +
 labs(x = "Age of respondent",
 y = "Number of respondents",
 fill = "Voted for") +
 scale_fill_brewer(palette = "Set1")

viridis <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar() +
 theme_minimal() +
 labs(x = "Age of respondent",
 y = "Number of respondents",
 fill = "Voted for") +
 scale_fill_viridis(discrete = TRUE)

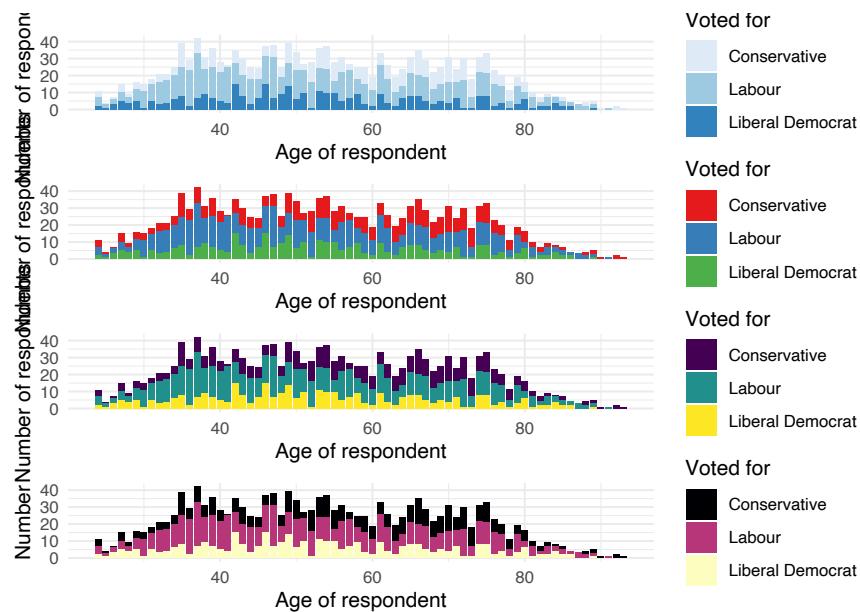
viridismagma <-
 beps |>
 ggplot(mapping = aes(x = age, fill = vote)) +
 geom_bar() +
 theme_minimal() +
 labs(x = "Age of respondent",
```

```

y = "Number",
 fill = "Voted for") +
 scale_fill_viridis(discrete = TRUE,
 option = "magma")

RColorBrewerBrBG /
 RColorBrewerSet2 /
 viridis /
 viridismagma

```



**FIGURE 6.11:** Distribution of age and vote preference, in the 1997-2001 British Election Panel Study

Details of the variety of palettes available in `RColorBrewer` and `viridis` are available in their help files. Many different palettes are available, and we can also build our own. That said, color is something to be considered with a great deal of care and it should only be added to increase the amount of information that is communicated (Cleveland, 1994). Colors should not be added to graphs unnecessarily—that is to say, they must play some role. Typically, that role is to distinguish different groups, and that implies making the colors dissimilar. Colors may also be appropriate if there is some relationship between the color and the variable, for instance if making a graph of sales of, say, mangoes and raspberries, it could help the reader if the colors were yellow and red, respectively (Franconeri et al., 2021, p. 121).

### 6.2.2 Scatterplots

We are often interested in the relationship between two variables. We can use scatterplots to show this information. Unless there is a good reason to move to a different option, a scatterplot is almost always the best choice (Weissgerber et al., 2015). Indeed, ‘among all forms of statistical graphics, the scatterplot may be considered the most versatile and generally useful invention in the entire history of statistical graphics.’ (Friendly and Wainer, 2021, p. 121) To illustrate scatterplots, we use `WDI` (Arel-Bundock, 2021b) to download some economic indicators from the World Bank.

---

**Oh, you think we have good data on that!** Gross Domestic Product (GDP) ‘combines in a single figure, and with no double counting, all the output (or production) carried out by all the firms, non-profit institutions, government bodies and households in a given country during a given period, regardless of the type of goods and services produced, provided that the production takes place within the country’s economic territory’ (OECD (2014), p. 15). The modern concept was developed by Simon Kuznets and is widely used and reported. There is a certain comfort in having a definitive and concrete single number to describe something as complicated as the entire economic activity of a country. And it is crucial that we have such summary statistics. But as with any summary statistic, its strength is also its weakness. A single number necessarily loses information about constituent components, and these distributional differences are critical. It highlights short term economic progress over longer term improvements. And ‘the quantitative definiteness of the estimates makes it easy to forget their dependence upon imperfect data and the consequently wide margins of possible error to which both totals and components are liable’ (Kuznets, 1941, p. xxvi). Reliance on any one summary measure of economic performance presents a misguided picture not only of a country’s economy, but also of its peoples.

---

```
install.packages('WDI')
```

```
library(tidyverse)
```

```

library(WDI)
WDIsearch("gdp growth")
#> indicator
#> [1,] "5.51.01.10.gdp"
#> [2,] "6.0.GDP_growth"
#> [3,] "NV.AGR.TOTL.ZG"
#> [4,] "NY.GDP.MKTP.KD.ZG"
#> [5,] "NY.GDP.MKTP.KN.87.ZG"
#> name
#> [1,] "Per capita GDP growth"
#> [2,] "GDP growth (annual %)"
#> [3,] "Real agricultural GDP growth rates (%)"
#> [4,] "GDP growth (annual %)"
#> [5,] "GDP growth (annual %)"
WDIsearch("inflation")
#> indicator
#> [1,] "FP.CPI.TOTL.ZG"
#> [2,] "FP.FPI.TOTL.ZG"
#> [3,] "FP.WPI.TOTL.ZG"
#> [4,] "NY.GDP.DEFL.87.ZG"
#> [5,] "NY.GDP.DEFL.KD.ZG"
#> [6,] "NY.GDP.DEFL.KD.ZG.AD"
#> name
#> [1,] "Inflation, consumer prices (annual %)"
#> [2,] "Inflation, food prices (annual %)"
#> [3,] "Inflation, wholesale prices (annual %)"
#> [4,] "Inflation, GDP deflator (annual %)"
#> [5,] "Inflation, GDP deflator (annual %)"
#> [6,] "Inflation, GDP deflator: linked series (annual %)"
WDIsearch("population, total")
#> indicator name
#> "SP.POP.TOTL" "Population, total"
WDIsearch("Unemployment, total")
#> indicator
#> [1,] "SL.UEM.TOTL.NE.ZS"
#> [2,] "SL.UEM.TOTL.ZS"
#> name
#> [1,] "Unemployment, total (% of total labor force) (national estimate)"
#> [2,] "Unemployment, total (% of total labor force) (modeled ILO estimate)"

```

```

world_bank_data <-
 WDI(indicator = c("FP.CPI.TOTL.ZG",
 "NY.GDP.MKTP.KD.ZG",

```

```

 "SP.POP.TOTL",
 "SL.UEM.TOTL.NE.ZS"
),
country = c("AU", "ET", "IN", "US")
)

```

At this point we may like to change the names to be more meaningful and only keep the columns that we need.

```

world_bank_data <-
 world_bank_data |>
 rename(inflation = FP.CPI.TOTL.ZG,
 gdp_growth = NY.GDP.MKTP.KD.ZG,
 population = SP.POP.TOTL,
 unemployment_rate = SL.UEM.TOTL.NE.ZS
) |>
 select(-iso2c)

head(world_bank_data)
#> # A tibble: 6 x 6
#> country year inflation gdp_growth population
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 Australia 1960 3.73 NA 10276477
#> 2 Australia 1961 2.29 2.48 10483000
#> 3 Australia 1962 -0.319 1.29 10742000
#> 4 Australia 1963 0.641 6.21 10950000
#> 5 Australia 1964 2.87 6.98 11167000
#> 6 Australia 1965 3.41 5.98 11388000
#> # ... with 1 more variable: unemployment_rate <dbl>

```

Now let us look at income as a function of years of education (Figure 6.12).

```

world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_point()
#> Warning: Removed 26 rows containing missing values
#> (geom_point).

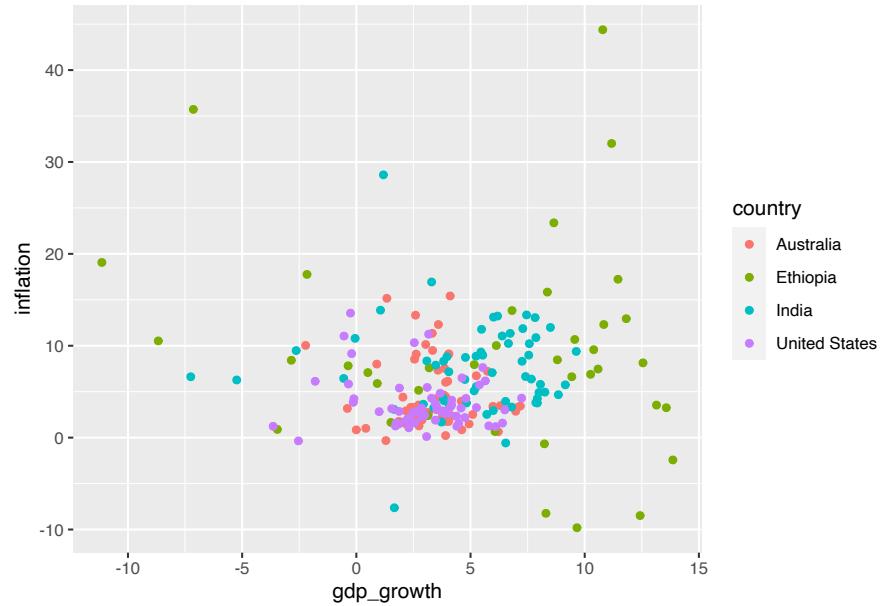
```

As with the bar plots, we change the theme, and update the labels (Figure 6.13).

```

world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +

```



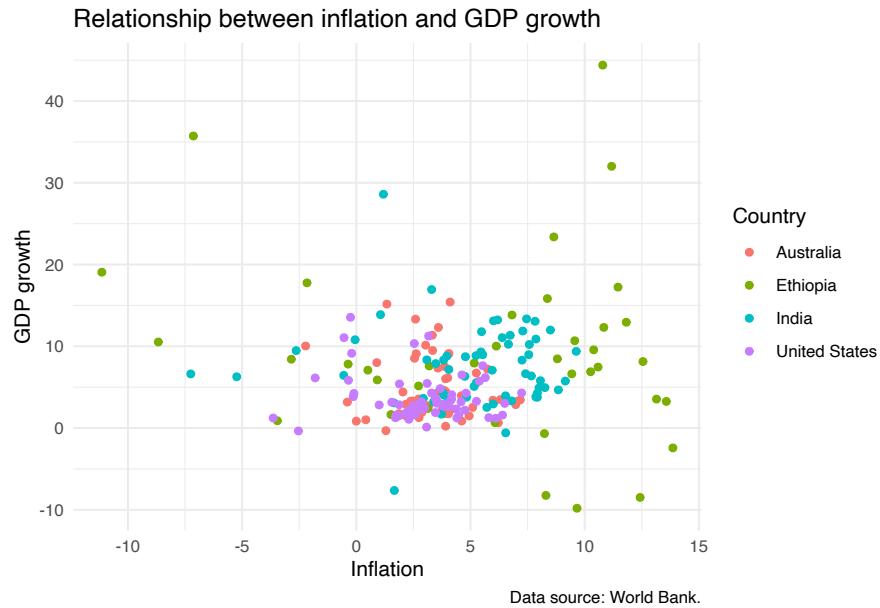
**FIGURE 6.12:** Relationship between inflation and GDP for Australia, Ethiopia, India, and the US

```
geom_point() +
theme_minimal() +
labs(x = "Inflation",
y = "GDP growth",
color = "Country",
title = "Relationship between inflation and GDP growth",
caption = "Data source: World Bank.")
#> Warning: Removed 26 rows containing missing values
#> (geom_point).
```

We use ‘color’ instead of ‘fill’ because we are using dots rather than bars. This also then slightly affects how we change the palette (Figure 6.14).

```
library(patchwork)

RColorBrewerBrBG <-
world_bank_data |>
ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
geom_point() +
```



**FIGURE 6.13:** Relationship between inflation and GDP for Australia, Ethiopia, India, and the US

```

theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.") +
 scale_color_brewer(palette = "Blues")

RColorBrewerSet2 <-
 world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_point() +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.") +
 scale_color_brewer(palette = "Set1")

```

```

viridis <-
 world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_point() +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.") +
 scale_colour_viridis_d()

viridismagma <-
 world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_point() +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.") +
 scale_colour_viridis_d(option = "magma")

(RColorBrewerBrBG + RColorBrewerSet2) /
 (viridis + viridismagma)

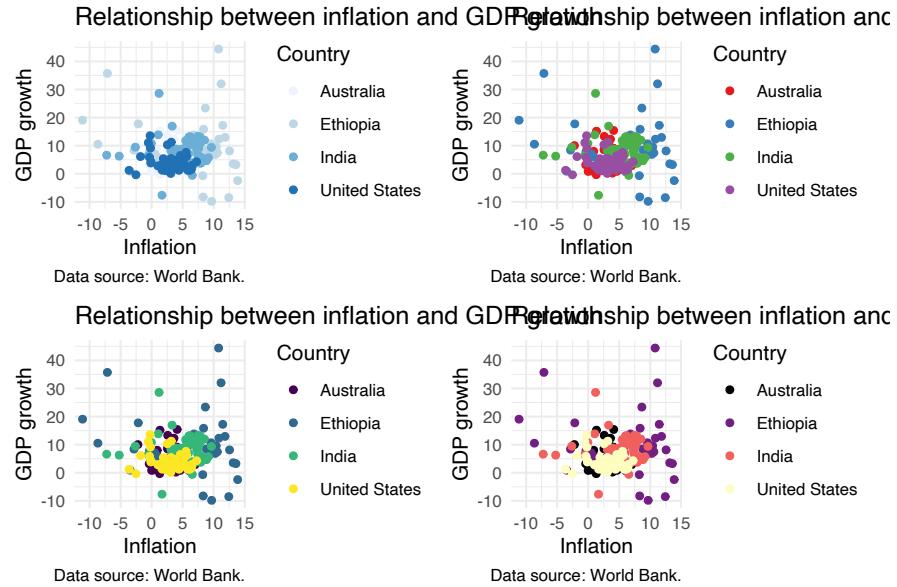
```

The dots of a dot plot often overlap. We can address this situation in one of two ways: adding a degree of transparency to our dots with ‘alpha’ (Figure 6.15). The value for ‘alpha’ can vary between 0, which is fully transparent, and 1, which is completely opaque. We can also specify a small amount by which we are comfortable if the points move with `geom_jitter()` (Figure 6.16). We can specify which direction movement occurs with ‘width’ or ‘height’. The decision between these two options turns on the degree to which exact accuracy matters, and the number of points.

```

world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_point(alpha = 0.5) +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",

```

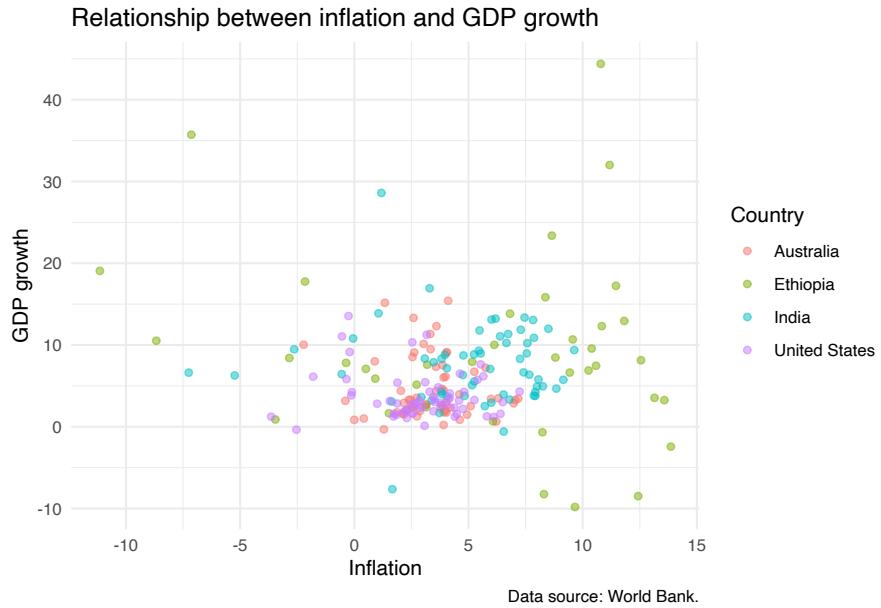


**FIGURE 6.14:** Relationship between inflation and GDP for Australia, Ethiopia, India, and the US

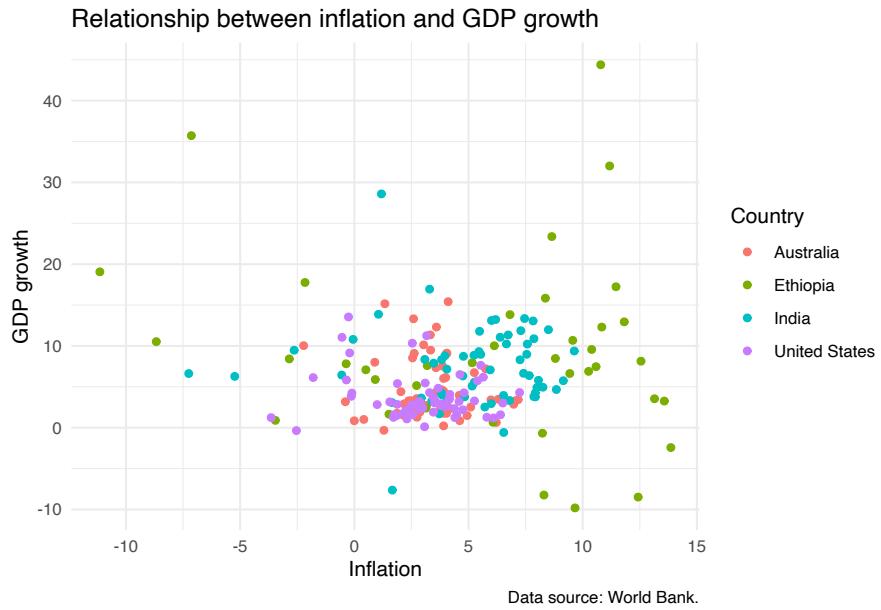
```
title = "Relationship between inflation and GDP growth",
caption = "Data source: World Bank.")
#> Warning: Removed 26 rows containing missing values
#> (geom_point).
```

```
world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_jitter() +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.")
#> Warning: Removed 26 rows containing missing values
#> (geom_point).
```

A common use case for a scatterplot is to illustrate a relationship between two variables. It can be useful to add a line of best fit using `geom_smooth()` (Fig-



**FIGURE 6.15:** Relationship between inflation and GDP for Australia, Ethiopia, India, and the US



**FIGURE 6.16:** Relationship between inflation and GDP for Australia, Ethiopia, India, and the US

ure 6.17). By default `geom_smooth()` will impose a X relationship. By default, loess smoothing is used for datasets with less than 1,000 observations, but we can specify the relationship using ‘method’, change the color with ‘color’ and remove standard errors with ‘se’. We use `geom_smooth()` to add a layer to the graph, and so it inherits all the settings that it can from `ggplot()`. For instance, that is why here we have one line for each country. We could overwrite that by specifying a particular color, in which case we would only have one line.

```
defaults <-
 world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_jitter() +
 geom_smooth() +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.")

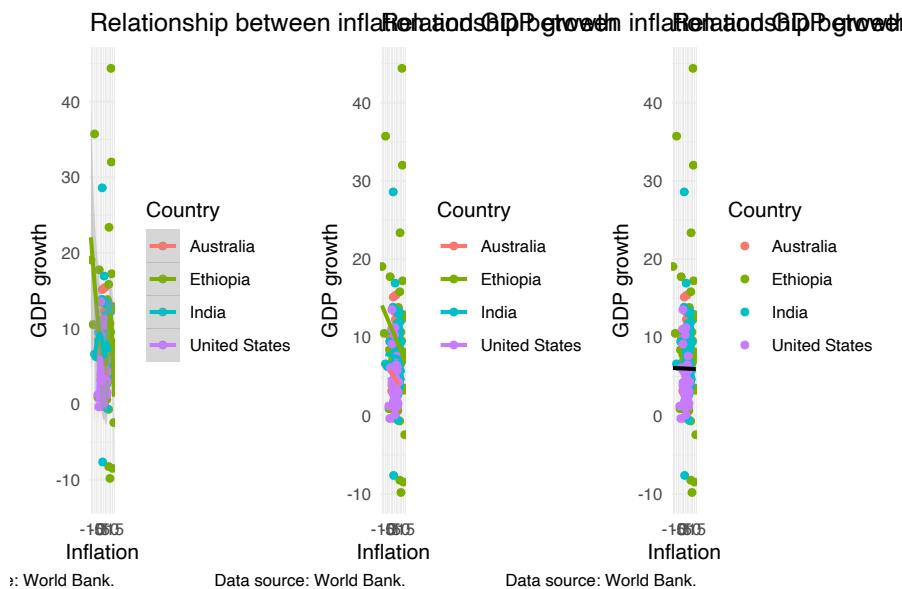
straightline <-
 world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_jitter() +
 geom_smooth(method = lm, se = FALSE) +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.")

onestraightline <-
 world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, y = inflation, color = country)) +
 geom_jitter() +
 geom_smooth(method = lm, color = "black", se = FALSE) +
 theme_minimal() +
 labs(x = "Inflation",
 y = "GDP growth",
 color = "Country",
 title = "Relationship between inflation and GDP growth",
 caption = "Data source: World Bank.")
```

```
(defaults + straightline + onestraightline)
#> `geom_smooth()` using method = 'loess' and formula 'y ~ x'
#> Warning: Removed 26 rows containing non-finite values
#> (stat_smooth).
#> Warning: Removed 26 rows containing missing values
#> (geom_point).
#> `geom_smooth()` using formula 'y ~ x'
#> Warning: Removed 26 rows containing non-finite values
#> (stat_smooth).

#> Warning: Removed 26 rows containing missing values (geom_point).
#> `geom_smooth()` using formula 'y ~ x'
#> Warning: Removed 26 rows containing non-finite values
#> (stat_smooth).

#> Warning: Removed 26 rows containing missing values (geom_point).
```

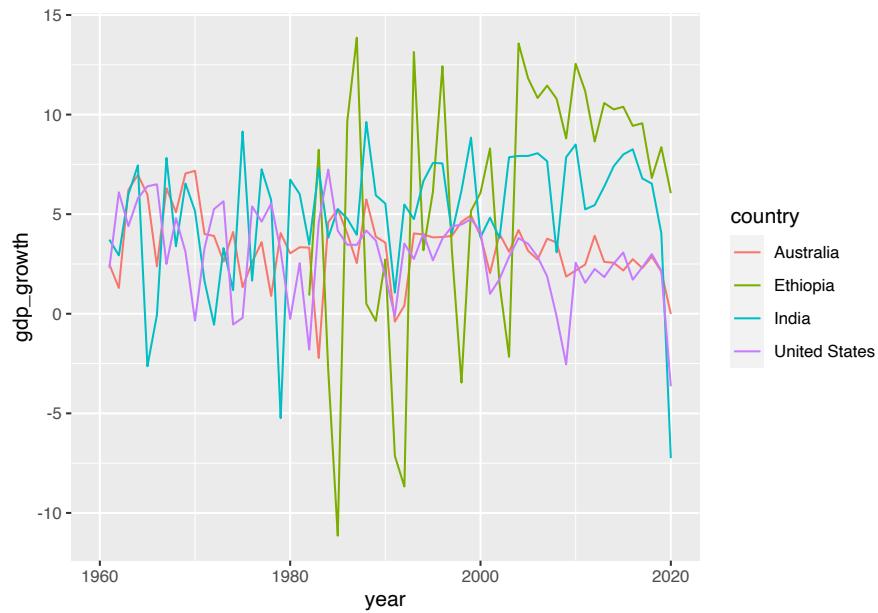


**FIGURE 6.17:** Relationship between inflation and GDP for Australia, Ethiopia, India, and the US

### 6.2.3 Line plots

We can use a line plot when we have variables that should be joined together, for instance, economic time series. We will continue with the dataset from the World Bank and focus on initially on GDP (Figure 6.18).

```
world_bank_data |>
 ggplot(mapping = aes(x = year, y = gdp_growth, color = country)) +
 geom_line()
#> Warning: Removed 25 row(s) containing missing values
#> (geom_path).
```



**FIGURE 6.18:** GDP over time for Australia, Ethiopia, India, and the US

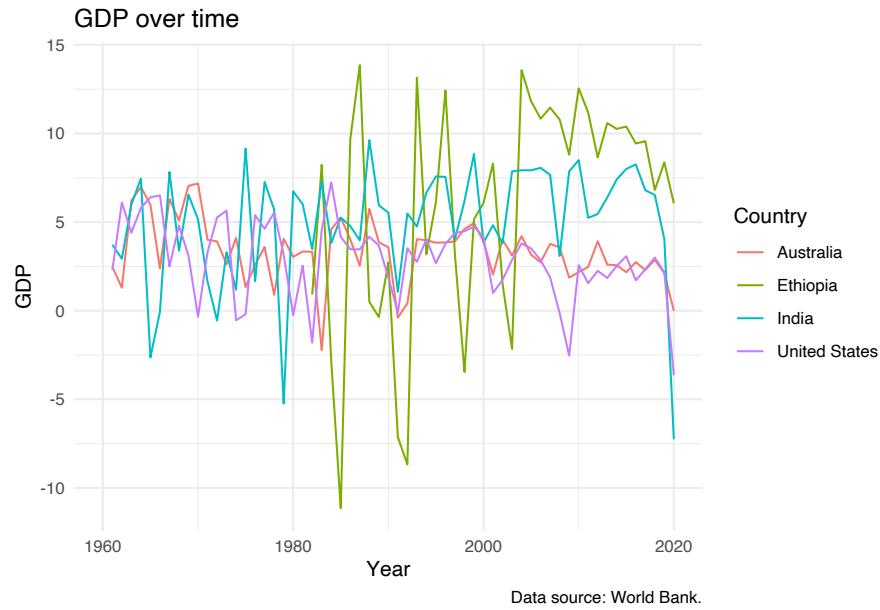
As before, we can adjust the theme and labels (Figure 6.19).

```
world_bank_data |>
 ggplot(mapping = aes(x = year, y = gdp_growth, color = country)) +
 geom_line() +
 theme_minimal() +
 labs(x = "Year",
 y = "GDP",
 color = "Country",
 title = "GDP over time",
```

```

 caption = "Data source: World Bank."
#> Warning: Removed 25 row(s) containing missing values
#> (geom_path).

```



**FIGURE 6.19:** GDP over time for Australia, Ethiopia, India, and the US

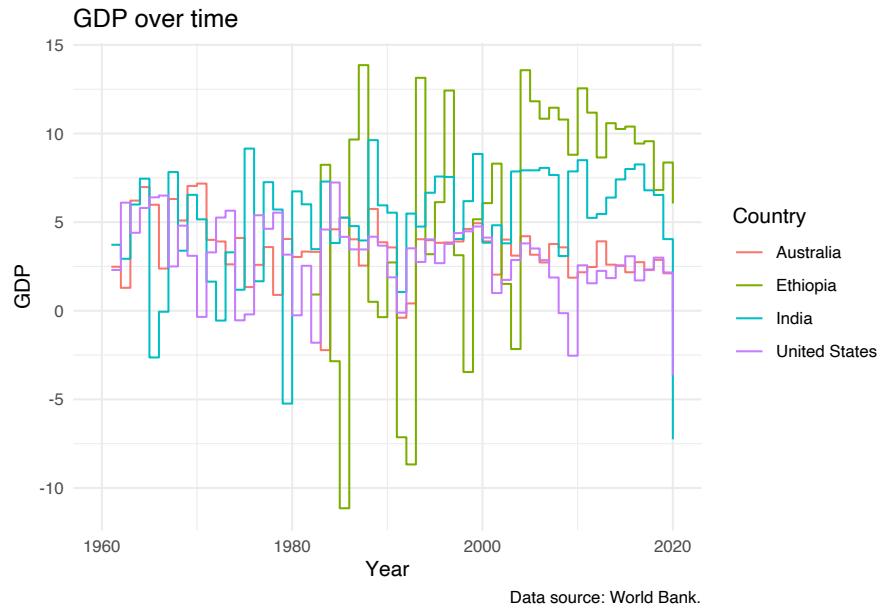
We can use a slight variant, `geom_step()` to focus attention on the change from year to year (Figure 6.20).

```

world_bank_data |>
 ggplot(mapping = aes(x = year, y = gdp_growth, color = country)) +
 geom_step() +
 theme_minimal() +
 labs(x = "Year",
 y = "GDP",
 color = "Country",
 title = "GDP over time",
 caption = "Data source: World Bank.")
#> Warning: Removed 25 row(s) containing missing values
#> (geom_path).

```

The Phillips curve is the name given to plot of the relationship between unemployment and inflation over time. An inverse relationship is sometimes found



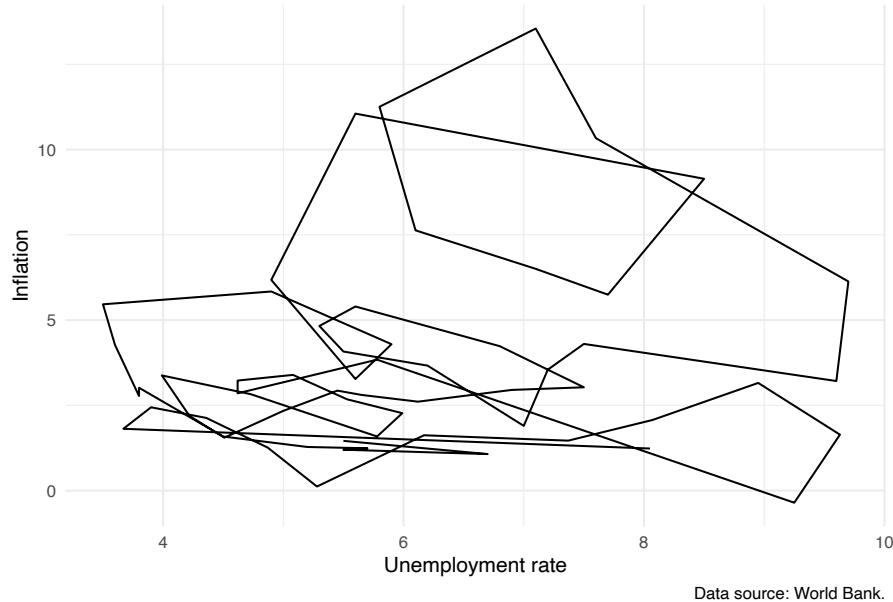
**FIGURE 6.20:** GDP over time for Australia, Ethiopia, India, and the US

in the data, for instance in the UK between 1861 and 1957 (Phillips, 1958). We have a variety of ways to investigate this including `geom_path()` which links values in the order they appear in the dataset. In Figure 6.21 we show a Phillips curve for the US between 1960 and 2020, and it is clear that any relationship that once existed between these variables does not appear to still exist.

```
world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = unemployment_rate, y = inflation)) +
 geom_path() +
 theme_minimal() +
 labs(x = "Unemployment rate",
 y = "Inflation",
 caption = "Data source: World Bank.")
```

#### 6.2.4 Histograms

A histogram is useful to show the shape of a continuous variable and works by constructing counts of the number of observations in different subsets of



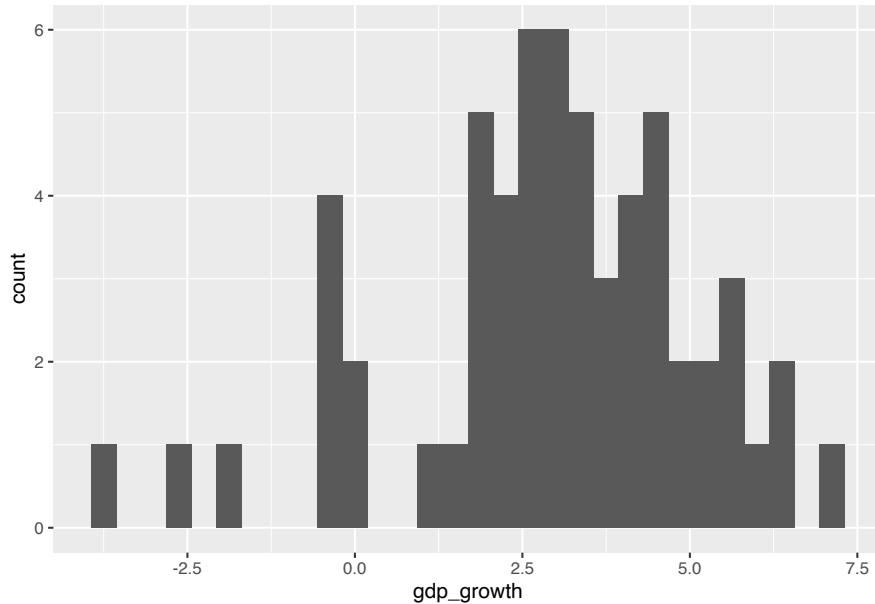
**FIGURE 6.21:** Phillips curve for the US (1960-2020)

the support, called ‘bins’. In Figure 6.22 we examine the distribution of GDP in the US.

```
world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram()
```

And again we can add a theme and labels (Figure 6.23).

```
world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram() +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
```



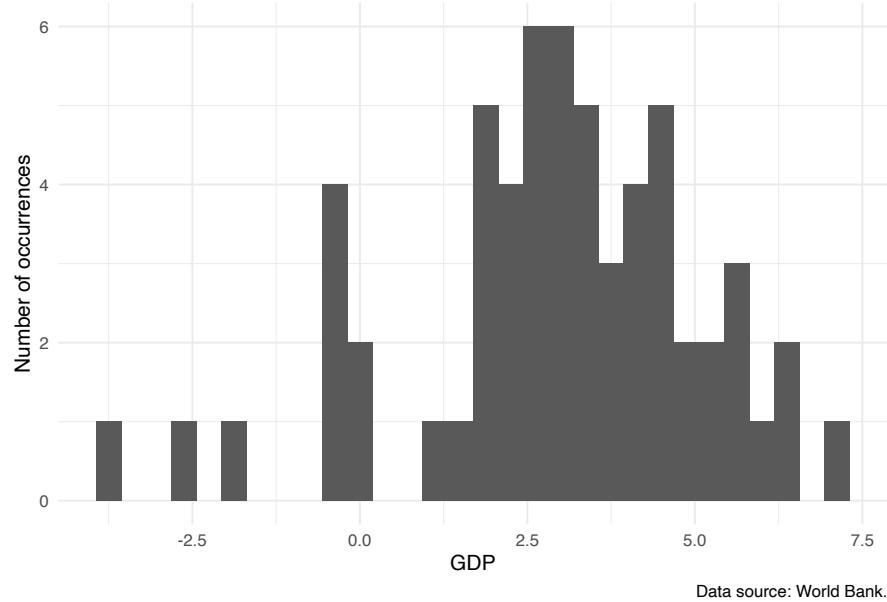
**FIGURE 6.22:** Distribution of income

```
#> Warning: Removed 1 rows containing non-finite values
#> (stat_bin).
```

The key component determining the shape of a histogram is the number of bins. This can be specified in one of two ways: 1) specifying the number of ‘bins’ to include, or 2) specifying how wide they should be with ‘binwidth’ (Figure 6.24).

```
twobins <-
 world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram(bins = 2) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")
```

```
fivebins <-
 world_bank_data |>
```



**FIGURE 6.23:** Distribution of GDP in the United States

```

filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram(bins = 5) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")

twentybins <-
 world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram(bins = 20) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")

halfbinwidth <-
 world_bank_data |>
 filter(country == "United States") |>

```

```

ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram(binwidth = 0.5) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")

twobinwidth <-
 world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram(binwidth = 2) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")

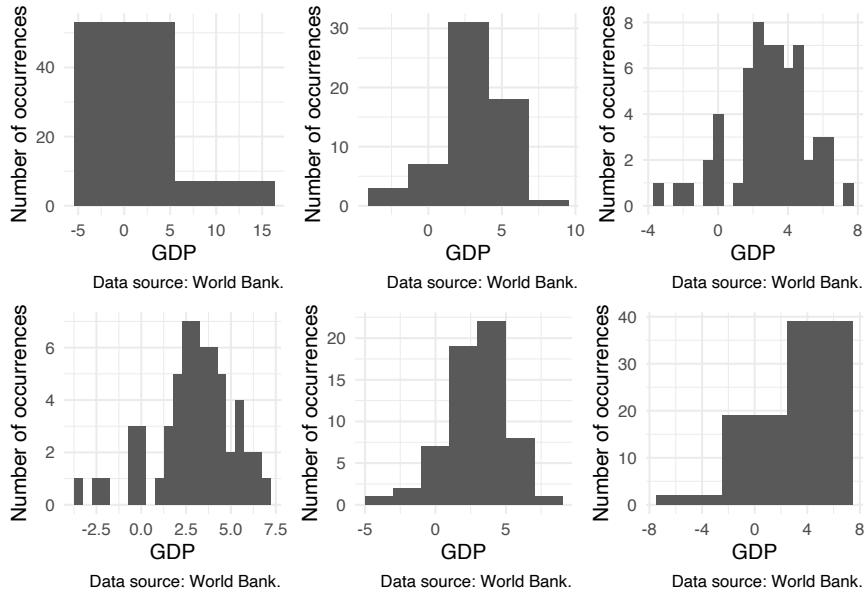
fivebinwidth <-
 world_bank_data |>
 filter(country == "United States") |>
 ggplot(mapping = aes(x = gdp_growth)) +
 geom_histogram(binwidth = 5) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 caption = "Data source: World Bank.")

(twobins + fivebins + twentybins) / (halfbinwidth + twobinwidth + fivebinwidth)

```

The histogram is smoothing the data, and the number of bins affects how much smoothing occurs. When there are only two bins then the data are very smooth, but we have lost a great deal of accuracy. More specifically, ‘the histogram estimator is a piecewise constant function where the height of the function is proportional to the number of observations in each bin’ (Wasserman, 2005, p. 303). Too few bins result in a biased estimator, while too many bins results in an estimator with high variance. Our decision as to the number of bins, or their width, is concerned with trying to balance bias and variance. This will depend on a variety of concerns including the subject matter and the goal (Cleveland, 1994, p. 135).

Finally, while we can use ‘fill’ to distinguish between different types of observations, it can get quite messy. It is usually better to give away showing the distribution with columns and instead trace the outline of the distribu-

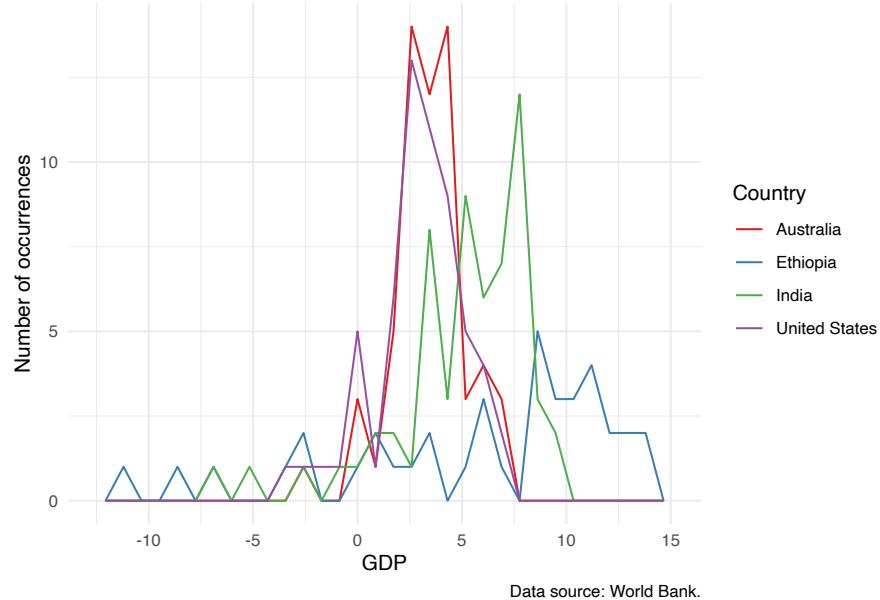


**FIGURE 6.24:** Distribution of GDP in the United States

tion, using `geom_freqpoly()` (Figure 6.25) or to build it up using dots with `geom_dotplot()` (Figure 6.26) .

```
world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, color = country)) +
 geom_freqpoly() +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 color = "Country",
 caption = "Data source: World Bank.") +
 scale_color_brewer(palette = "Set1")
```

```
world_bank_data |>
 ggplot(mapping = aes(x = gdp_growth, group = country, fill = country)) +
 geom_dotplot(method = 'histodot', alpha = 0.4) +
 theme_minimal() +
 labs(x = "GDP",
 y = "Number of occurrences",
 fill = "Country",
```



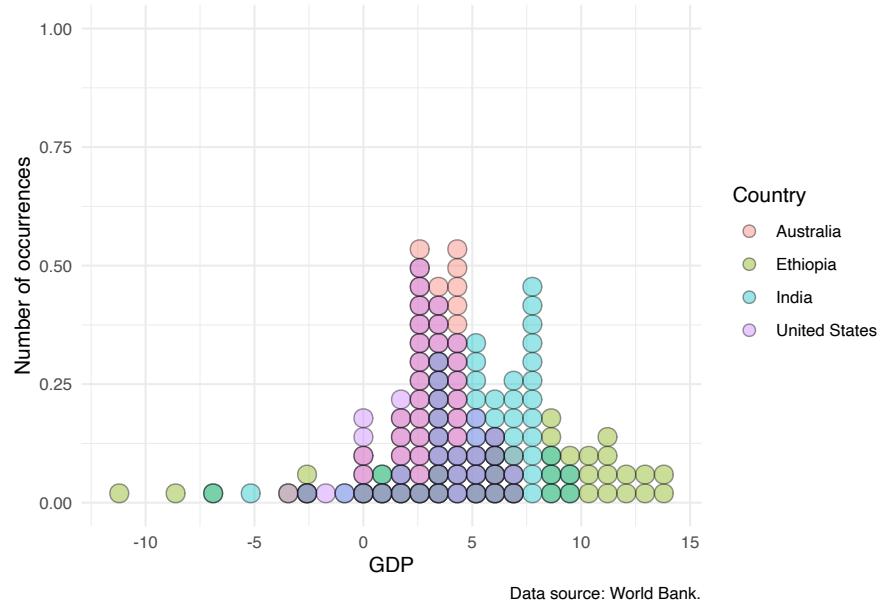
**FIGURE 6.25:** Distribution of GDP in the United States

```
caption = "Data source: World Bank.") +
scale_color_brewer(palette = "Set1")
```

### 6.2.5 Boxplots

Boxplots are almost never an appropriate choice because they hide the distribution of data, rather than show it. Unless we need to compare the summary statistics of many variables at once, then they should almost never be used. This is because the same boxplot can apply to very different distributions. To see this, consider some simulated data from the beta distribution of two types. One type of data contains draws from two beta distributions: one that is right skewed and another that is left skewed. The other type of data contains draws from a beta distribution with no skew.

```
set.seed(853)
both_left_and_right_skew <-
 c(
 rbeta(500, 5, 2),
 rbeta(500, 2, 5)
)
```



**FIGURE 6.26:** Distribution of GDP in the United States

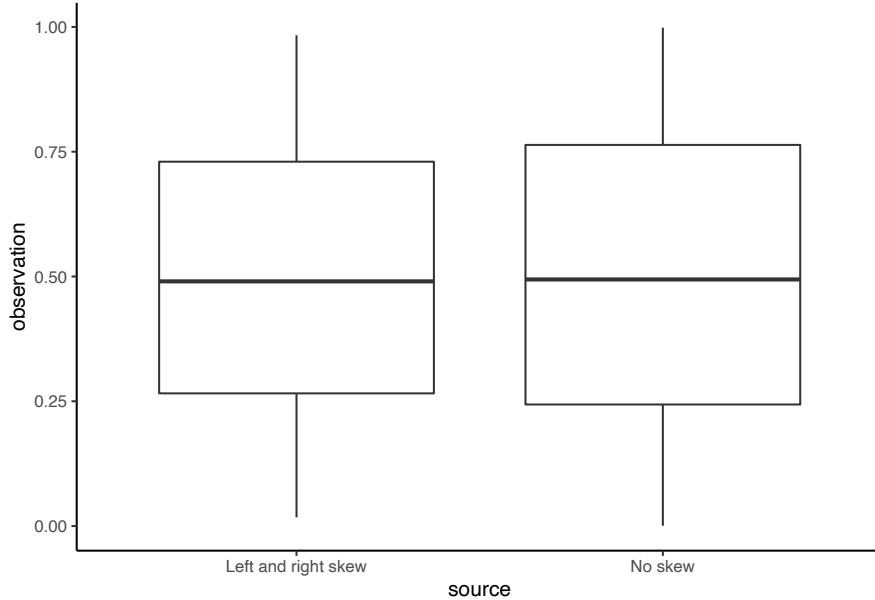
```
no_skew <-
 rbeta(1000, 1, 1)

beta_distributions <-
 tibble(
 observation = c(both_left_and_right_skew, no_skew),
 source = c(rep("Left and right skew", 1000),
 rep("No skew", 1000)
)
)
```

We can first compare the boxplots of the two series (Figure 6.27).

```
beta_distributions |>
 ggplot(aes(x = source, y = observation)) +
 geom_boxplot() +
 theme_classic()
```

But if we plot the actual data then we can see how different they are (Figure 6.28).



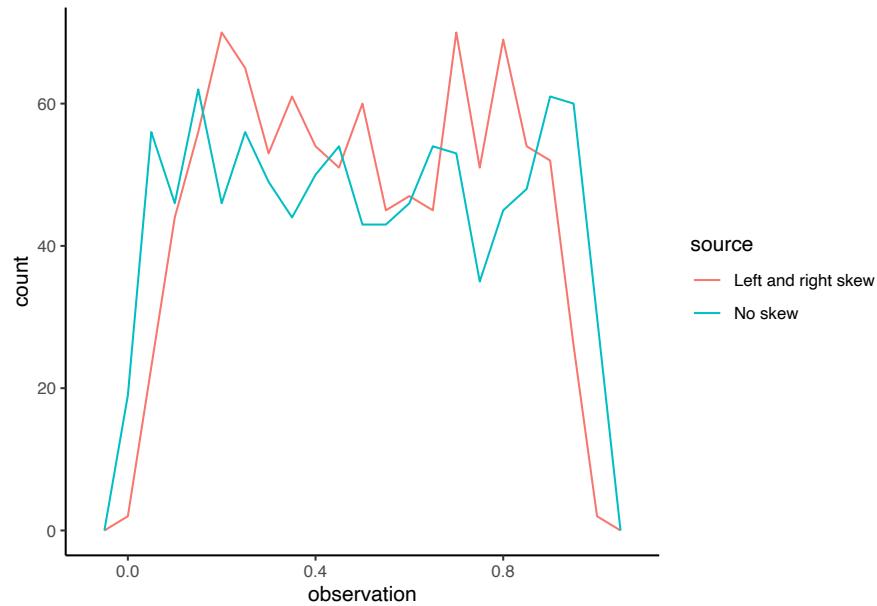
**FIGURE 6.27:** Data drawn from beta distributions with different parameters

```
beta_distributions |>
 ggplot(aes(x = observation, color = source)) +
 geom_freqpoly(binwidth = 0.05) +
 theme_classic()
```

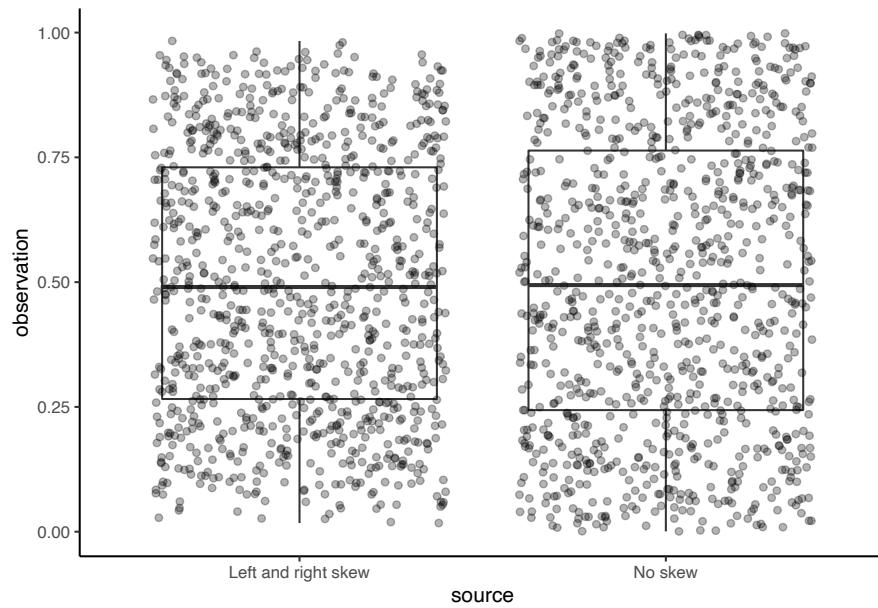
One way forward, if a boxplot must be included, is to include the actual data as a layer on top of the boxplot (Figure 6.29).

```
beta_distributions |>
 ggplot(aes(x = source, y = observation)) +
 geom_boxplot() +
 geom_jitter(alpha = 0.3) +
 theme_classic()
```

An even better solution is to graph the quantiles of each distribution against (Figure 6.30). The Q-Q plot was developed by Wilk and Gnanadesikan (1968) and requires us to plot the distributions against each other.

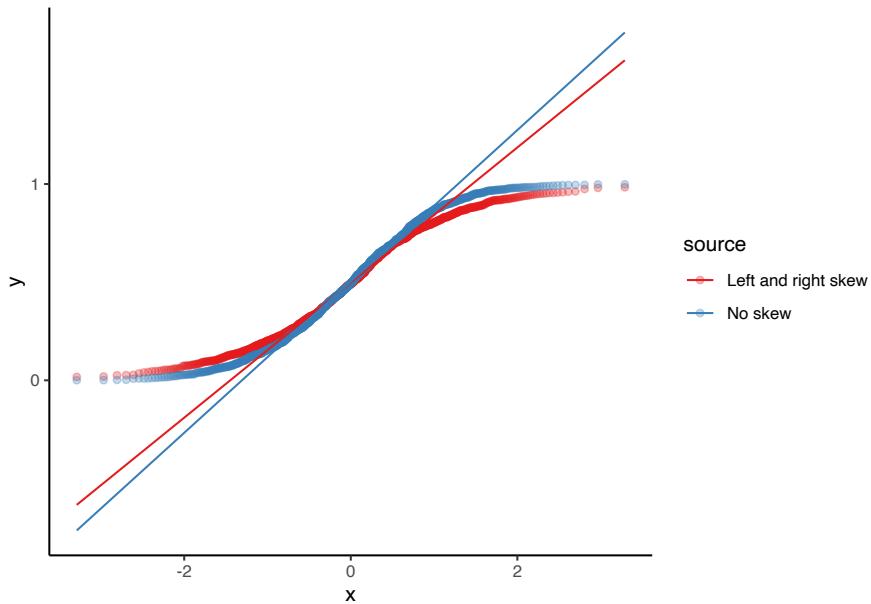


**FIGURE 6.28:** Data drawn from beta distributions with different parameters



**FIGURE 6.29:** Data drawn from beta distributions with different parameters

```
beta_distributions |>
 ggplot(aes(sample = observation, color = source)) +
 stat_qq(alpha = 0.3) +
 stat_qq_line() +
 theme_classic() +
 scale_color_brewer(palette = "Set1")
```



**FIGURE 6.30:** Data drawn from beta distributions with different parameters

---

### 6.3 Tables

Tables are critical for telling a compelling story. Tables can communicate less information than a graph, but they can do so at a high fidelity. We primarily use tables in three ways:

1. To show some of our actual dataset, for which we use `kable()` from `knitr` (Xie, 2021), alongside `kableExtra` (Zhu, 2020).
2. To communicate summary statistics, for which we use `gt` (Iannone et al., 2020) and `modelsummary` (Arel-Bundock, 2021a).

3. To display regression results, for which we use `modelsummary` ([Arell-Bundock, 2021a](#)).

### 6.3.1 Showing part of a dataset

We will illustrate showing part of a dataset using `kable()` from `knitr` and drawing on `kableExtra` for enhancement. We will use the World Bank dataset on inflation and GDP from earlier.

```
library(knitr)
head(world_bank_data)
#> # A tibble: 6 x 6
#> country year inflation gdp_growth population
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 Australia 1960 3.73 NA 10276477
#> 2 Australia 1961 2.29 2.48 10483000
#> 3 Australia 1962 -0.319 1.29 10742000
#> 4 Australia 1963 0.641 6.21 10950000
#> 5 Australia 1964 2.87 6.98 11167000
#> 6 Australia 1965 3.41 5.98 11388000
#> # ... with 1 more variable: unemployment_rate <dbl>
```

To begin, we can display the first ten rows with the default `kable()` settings.

```
world_bank_data |>
 slice(1:10) |>
 kable()
```

country	year	inflation	gdp_growth	population	unemployment_rate
Australia	1960	3.7288136	NA	10276477	NA
Australia	1961	2.2875817	2.483271	10483000	NA
Australia	1962	-0.3194888	1.294468	10742000	NA
Australia	1963	0.6410256	6.214949	10950000	NA
Australia	1964	2.8662420	6.978540	11167000	NA
Australia	1965	3.4055728	5.980893	11388000	NA
Australia	1966	3.2934132	2.381966	11651000	NA
Australia	1967	3.4782609	6.303650	11799000	NA
Australia	1968	2.5210084	5.095103	12009000	NA
Australia	1969	3.2786885	7.043526	12263000	NA

In order to be able to cross-reference it in text, we need to add a caption with ‘caption’. We can also make the column names more informative with ‘col.names’ and specify the number of digits to be displayed (Table 6.3).

**TABLE 6.3:** First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US

Country	Year	Inflation	GDP growth	Population	Unemployment rate
Australia	1960	3.7	NA	10276477	NA
Australia	1961	2.3	2.5	10483000	NA
Australia	1962	-0.3	1.3	10742000	NA
Australia	1963	0.6	6.2	10950000	NA
Australia	1964	2.9	7.0	11167000	NA
Australia	1965	3.4	6.0	11388000	NA
Australia	1966	3.3	2.4	11651000	NA
Australia	1967	3.5	6.3	11799000	NA
Australia	1968	2.5	5.1	12009000	NA
Australia	1969	3.3	7.0	12263000	NA

```
world_bank_data |>
 slice(1:10) |>
 kable(
 caption = "First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, ...",
 col.names = c("Country", "Year", "Inflation", "GDP growth", "Population", "Unemployment rate"),
 digits = 1
)
```

When producing PDFs, the ‘booktabs’ option makes a host of small changes to the default display and results in tables that look better (Table 6.4). When using ‘booktabs’ we additionally should specify ‘linesep’ otherwise `kable()` adds a small space every five lines. (None of this will show up for html output.)

```
world_bank_data |>
 slice(1:10) |>
 kable(
 caption = "First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, ...",
 col.names = c("Country", "Year", "Inflation", "GDP growth", "Population", "Unemployment rate"),
 digits = 1,
 booktabs = TRUE,
 linesep = ""
)
```

```
world_bank_data |>
 slice(1:10) |>
 kable(
```

**TABLE 6.4:** First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US

Country	Year	Inflation	GDP growth	Population	Unemployment rate
Australia	1960	3.7	NA	10276477	NA
Australia	1961	2.3	2.5	10483000	NA
Australia	1962	-0.3	1.3	10742000	NA
Australia	1963	0.6	6.2	10950000	NA
Australia	1964	2.9	7.0	11167000	NA
Australia	1965	3.4	6.0	11388000	NA
Australia	1966	3.3	2.4	11651000	NA
Australia	1967	3.5	6.3	11799000	NA
Australia	1968	2.5	5.1	12009000	NA
Australia	1969	3.3	7.0	12263000	NA

**TABLE 6.5:** First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US

Country	Year	Inflation	GDP growth	Population	Unemployment rate
Australia	1960	3.7	NA	10276477	NA
Australia	1961	2.3	2.5	10483000	NA
Australia	1962	-0.3	1.3	10742000	NA
Australia	1963	0.6	6.2	10950000	NA
Australia	1964	2.9	7.0	11167000	NA
Australia	1965	3.4	6.0	11388000	NA
Australia	1966	3.3	2.4	11651000	NA
Australia	1967	3.5	6.3	11799000	NA
Australia	1968	2.5	5.1	12009000	NA
Australia	1969	3.3	7.0	12263000	NA

```

caption = "First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, ...
col.names = c("Country", "Year", "Inflation", "GDP growth", "Population", "Unemployment rate")
digits = 1,
booktabs = TRUE
)

```

We can specify the alignment of the columns using a character vector of ‘l’ (left), ‘c’ (centre), and ‘r’ (right) (Table 6.6). Additionally, (and this is not relevant for this table), we could specify groupings for numbers that are at least one thousand using ‘format.args = list(big.mark = “,”)’.

**TABLE 6.6:** First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US

Country	Year	Inflation	GDP growth	Population	Unemployment rate
Australia	1960	3.7	NA	10276477	NA
Australia	1961	2.3	2.5	10483000	NA
Australia	1962	-0.3	1.3	10742000	NA
Australia	1963	0.6	6.2	10950000	NA
Australia	1964	2.9	7.0	11167000	NA
Australia	1965	3.4	6.0	11388000	NA
Australia	1966	3.3	2.4	11651000	NA
Australia	1967	3.5	6.3	11799000	NA
Australia	1968	2.5	5.1	12009000	NA
Australia	1969	3.3	7.0	12263000	NA

```
world_bank_data |>
 slice(1:10) |>
 kable(
 caption = "First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US",
 col.names = c("Country", "Year", "Inflation", "GDP growth", "Population", "Unemployment rate"),
 digits = 1,
 booktabs = TRUE,
 linesep = "",
 align = c('l', 'l', 'c', 'c', 'r', 'r'),
)
```

We can use `kableExtra` (Zhu, 2020) to add extra functionality to `kable`. For instance, we could add a row that groups some of the columns (Table 6.6).

```
library(kableExtra)

world_bank_data |>
 slice(1:10) |>
 kable(
 caption = "First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US",
 col.names = c("Country", "Year", "Inflation", "GDP growth", "Population", "Unemployment rate"),
 digits = 1,
 booktabs = TRUE,
 linesep = "",
 align = c('l', 'l', 'c', 'c', 'r', 'r'),
) |>
 add_header_above(c(" " = 2, "Economic indicators" = 4))
```

**TABLE 6.7:** First ten rows of a dataset of economic indicators for Australia, Ethiopia, India, and the US

Country	Year	Economic indicators			
		Inflation	GDP growth	Population	Unemployment rate
Australia	1960	3.7	NA	10276477	NA
Australia	1961	2.3	2.5	10483000	NA
Australia	1962	-0.3	1.3	10742000	NA
Australia	1963	0.6	6.2	10950000	NA
Australia	1964	2.9	7.0	11167000	NA
Australia	1965	3.4	6.0	11388000	NA
Australia	1966	3.3	2.4	11651000	NA
Australia	1967	3.5	6.3	11799000	NA
Australia	1968	2.5	5.1	12009000	NA
Australia	1969	3.3	7.0	12263000	NA

	Unique (#)	Missing (%)	Mean	SD	Min	Median
year	61	0	1990.0	17.6	1960.0	1990.
inflation	238	3	6.1	6.3	-9.8	4.
gdp_growth	220	10	4.2	3.7	-11.1	3.
population	244	0	304 177 482.9	380 093 166.9	10 276 477.0	147 817 291.
unemployment_rate	104	52	6.0	1.9	1.2	5.

### 6.3.2 Communicating summary statistics

We can use `datasummary()` from `modelsummary` to create tables of summary statistics from our dataset.

```
library(modelsummary)

world_bank_data |>
 datasummary_skim()
```

By default it summarizes the ‘numeric’ variables, but we can ask for the ‘categorical’ variables (Table 6.8). Additionally we can add cross-references in the same way as `kable()`, that is, include a title and then cross-reference the name of the R chunk.

```
world_bank_data |>
```

**TABLE 6.8:** Summary of categorical variables for the inflation and GDP dataset

country	N	%
Australia	61	25.0
Ethiopia	61	25.0
India	61	25.0
United States	61	25.0

**TABLE 6.9:** Correlation between the variables for the inflation and GDP dataset

	year	inflation	gdp_growth	population	unemployment_rate
year	1	.	.	.	.
inflation	0.00	1	.	.	.
gdp_growth	0.10	0.00	1	.	.
population	0.24	0.07	0.15	1	.
unemployment_rate	-0.13	-0.14	-0.31	-0.35	1

```
datasummary_skim(type = "categorical",
 title = "Summary of categorical variables for the inflation and GDP dataset")
```

We can create a table that shows the correlation between variables using `datasummary_correlation()` (Table 6.9).

```
world_bank_data |>
 datasummary_correlation(
 title = "Correlation between the variables for the inflation and GDP dataset"
)
```

We typically need a table of descriptive statistics that we could add to our paper (Table 6.10). This is in contrast to Table 6.8 which would likely not be included in a paper. We can add a note about the source of the data using ‘notes’.

**TABLE 6.10:** Descriptive statistics for the inflation and GDP dataset

	Australia (N=61)		Ethiopia (N=61)		India (N=61)		U
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.	
year	1990.0	17.8	1990.0	17.8	1990.0	17.8	
inflation	4.7	3.8	8.7	10.4	7.4	5.0	
gdp_growth	3.4	1.8	5.9	6.4	5.0	3.3	
population	17244215.9	4328625.6	55662437.9	27626912.1	888774544.9	292997809.4	255
unemployment_rate	6.8	1.7	2.6	0.9	3.5	1.4	

Data source: World Bank.

Model 1	
(Intercept)	4.157 (0.352)
inflation	-0.002 (0.041)
Num.Obs.	218
R2	0.000
R2 Adj.	-0.005
AIC	1195.1
BIC	1205.3
Log.Lik.	-594.554
F	0.002

### 6.3.3 Display regression results

Finally, one common reason for needing a table is to report regression results. We will do this using `modelsummary()` from `modelsummary` (Arel-Bundock, 2021a).

```
first_model <- lm(formula = gdp_growth ~ inflation,
 data = world_bank_data)

modelsummary(first_model)
```

We can put a variety of different of different models together (Table 6.11).

```
second_model <- lm(formula = gdp_growth ~ inflation + country,
 data = world_bank_data)

third_model <- lm(formula = gdp_growth ~ inflation + country + population,
```

**TABLE 6.11:** Explaining GDP as a function of inflation

	Model 1	Model 2	Model 3
(Intercept)	4.157 (0.352)	3.728 (0.495)	3.668 (0.494)
inflation	-0.002 (0.041)	-0.075 (0.041)	-0.072 (0.041)
countryEthiopia		2.872 (0.757)	2.716 (0.758)
countryIndia		1.854 (0.655)	-0.561 (1.520)
countryUnited States		-0.524 (0.646)	-1.176 (0.742)
population			0.000 (0.000)
Num.Obs.	218	218	218
R2	0.000	0.110	0.123
R2 Adj.	-0.005	0.093	0.102
AIC	1195.1	1175.7	1174.5
BIC	1205.3	1196.0	1198.2
Log.Lik.	-594.554	-581.844	-580.266
F	0.002	6.587	5.939

```
data = world_bank_data)

modelsummary(list(first_model, second_model, third_model),
 title = "Explaining GDP as a function of inflation")
```

We can adjust the number of significant digits (Table 6.12).

```
modelsummary(list(first_model, second_model, third_model),
 fmt = 1,
 title = "Two models of GDP as a function of inflation")
```

---

## 6.4 Maps

In many ways maps can be thought of as another type of graph, where the x-axis is latitude, the y-axis is longitude, and there is some outline or a back-

**TABLE 6.12:** Two models of GDP as a function of inflation

	Model 1	Model 2	Model 3
(Intercept)	4.2 (0.4)	3.7 (0.5)	3.7 (0.5)
inflation	0.0 (0.0)	-0.1 (0.0)	-0.1 (0.0)
countryEthiopia		2.9 (0.8)	2.7 (0.8)
countryIndia		1.9 (0.7)	-0.6 (1.5)
countryUnited States		-0.5 (0.6)	-1.2 (0.7)
population		0.0 (0.0)	
Num.Obs.	218	218	218
R2	0.000	0.110	0.123
R2 Adj.	-0.005	0.093	0.102
AIC	1195.1	1175.7	1174.5
BIC	1205.3	1196.0	1198.2
Log.Lik.	-594.554	-581.844	-580.266
F	0.002	6.587	5.939

ground image. We have seen this type of set-up are used to this type of set-up, for instance, in the `ggplot2` setting, this is quite familiar.

```
ggplot() +
 geom_polygon(# First draw an outline
 data = some_data,
 aes(x = latitude,
 y = longitude,
 group = group
)) +
 geom_point(# Then add points of interest
 data = some_other_data,
 aes(x = latitude,
 y = longitude)
)
```

And while there are some small complications, for the most part it is as straight-forward as that. The first step is to get some data. There is some geographic data built into `ggplot2`, and there is additional information in `maps`.

```

library(maps)

canada <- map_data(database = "world", regions = "canada")
canadian_cities <- maps::canada.cities

head(canada)
#> long lat group order region subregion
#> 1 -59.78760 43.93960 1 1 Canada Sable Island
#> 2 -59.92227 43.90391 1 2 Canada Sable Island
#> 3 -60.03775 43.90664 1 3 Canada Sable Island
#> 4 -60.11426 43.93911 1 4 Canada Sable Island
#> 5 -60.11748 43.95337 1 5 Canada Sable Island
#> 6 -59.93604 43.93960 1 6 Canada Sable Island

head(canadian_cities)
#> name country.etc pop lat long capital
#> 1 Abbotsford BC 157795 49.06 -122.30 0
#> 2 Acton ON 8308 43.63 -80.03 0
#> 3 Acton Vale QC 5153 45.63 -72.57 0
#> 4 Airdrie AB 25863 51.30 -114.02 0
#> 5 Aklavik NT 643 68.22 -135.00 0
#> 6 Albanel QC 1090 48.87 -72.42 0

```

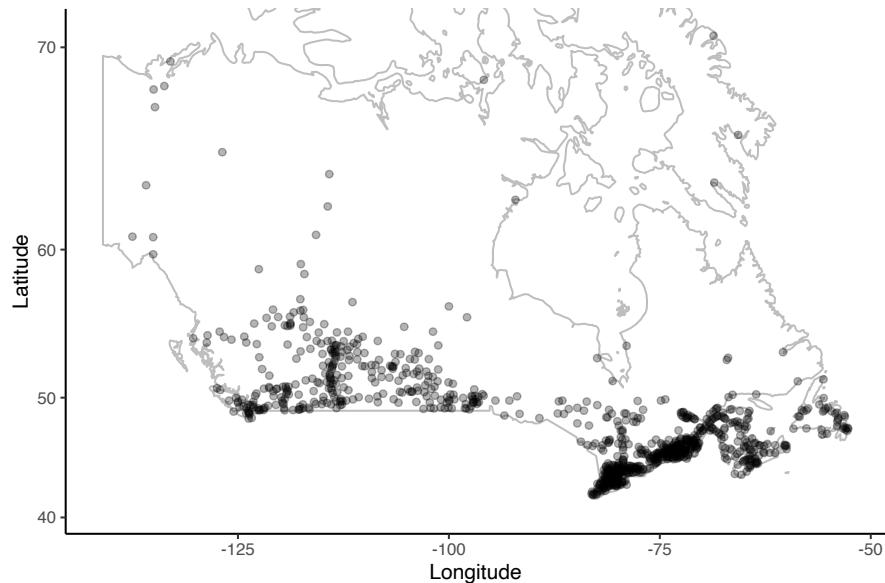
With that information in hand we can then create a map of Canada that shows the cities with a population over 1,000. (The `geom_polygon()` function within `ggplot2` draws shapes, by connecting points within groups. And the `coord_map()` function adjusts for the fact that we are making something that is 2D map to represent something that is 3D.)

```

ggplot() +
 geom_polygon(data = canada,
 aes(x = long,
 y = lat,
 group = group),
 fill = "white",
 colour = "grey") +
 coord_map(ylim = c(40, 70)) +
 geom_point(aes(x = canadian_cities$long,
 y = canadian_cities$lat),
 alpha = 0.3,
 color = "black") +
 theme_classic() +

```

```
labs(x = "Longitude",
 y = "Latitude")
```



As is often the case with R, there are many different ways to get started creating static maps. We have already seen how they can be built using only `ggplot2`, but `ggmap` brings additional functionality (Kahle and Wickham, 2013).

There are two essential components to a map:

- 1) some border or background image (also known as a tile); and
- 2) something of interest within that border or on top of that tile.

In `ggmap`, we use an open-source option for our tile, Stamen Maps: [maps.stamen.com](http://maps.stamen.com). And we use plot points based on latitude and longitude.

#### 6.4.1 Australian polling places

In Australia people go to specific locations, called booths, to vote. These booths have latitudes and longitudes and so we can plot these. One reason we may like to do this is to notice patterns over geographies.

To get started we need to get a tile. We are going to use `ggmap` to get a tile from Stamen Maps, which builds on OpenStreetMap ([openstreetmap.org](http://openstreetmap.org)). The main argument to this function is to specify a bounding box. This requires two latitudes - one for the top of the box and one for the bottom of the box -

and two longitudes - one for the left of the box and one for the right of the box. It can be useful to use Google Maps, or an alternative, to find the values of these that you need. The bounding box provides the coordinates of the edges that you are interested in. In this case we have provided it with coordinates such that it will be centered around Canberra, Australia, which is a small city that was created for the purposes of being the capital.

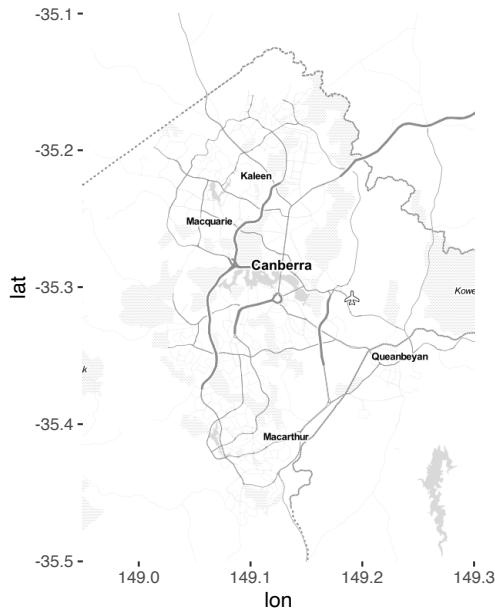
```
library(ggmap)

bbox <- c(left = 148.95, bottom = -35.5, right = 149.3, top = -35.1)
```

Once you have defined the bounding box, then the function `get_stamenmap()` will get the tiles in that area. The number of tiles that it needs to get depends on the zoom, and the type of tiles that it gets depends on the maptype. We have used a black-and-white type of map but the helpfile specifies others. At this point we can the map to maps to `ggmap()` and it will plot the tile! It will be actively downloading these tiles, and so it needs an internet connection.

```
canberra_stamen_map <- get_stamenmap(bbox, zoom = 11, maptype = "toner-lite")

ggmap(canberra_stamen_map)
```



Once we have a map then we can use `ggmap()` to plot it. (That circle in the middle of the map is where the Australian Parliament House is. Yes, the

Australian parliament is surrounded by circular roads, which Australians call ‘roundabouts’, actually Australians thought this was such a great idea, that we surrounded it by two of them.)

Now we want to get some data that we plot on top of our tiles. We will just plot the location of the polling places, based on which ‘division’ (the Australian equivalent to ‘ridings’ in Canada) it is. This is available here: <https://results.aec.gov.au/20499/Website/Downloads/HouseTppByPollingPlaceDownload-20499.csv>. The Australian Electoral Commission (AEC) is the official government agency that is responsible for elections in Australia.

```
Read in the booths data for each year
booths <-
 readr::read_csv(
 "https://results.aec.gov.au/24310/Website/Downloads/GeneralPollingPlacesDownload-24310.csv",
 skip = 1,
 guess_max = 10000
)

head(booths)
#> # A tibble: 6 x 15
#> State DivisionID DivisionNm PollingPlaceID
#> <chr> <dbl> <chr> <dbl>
#> 1 ACT 318 Bean 93925
#> 2 ACT 318 Bean 93927
#> 3 ACT 318 Bean 11877
#> 4 ACT 318 Bean 11452
#> 5 ACT 318 Bean 8761
#> 6 ACT 318 Bean 8763
#> # ... with 11 more variables: PollingPlaceTypeID <dbl>,
#> # PollingPlaceNm <chr>, PremisesNm <chr>,
#> # PremisesAddress1 <chr>, PremisesAddress2 <chr>,
#> # PremisesAddress3 <chr>, PremisesSuburb <chr>,
#> # PremisesStateAb <chr>, PremisesPostCode <chr>,
#> # Latitude <dbl>, Longitude <dbl>
```

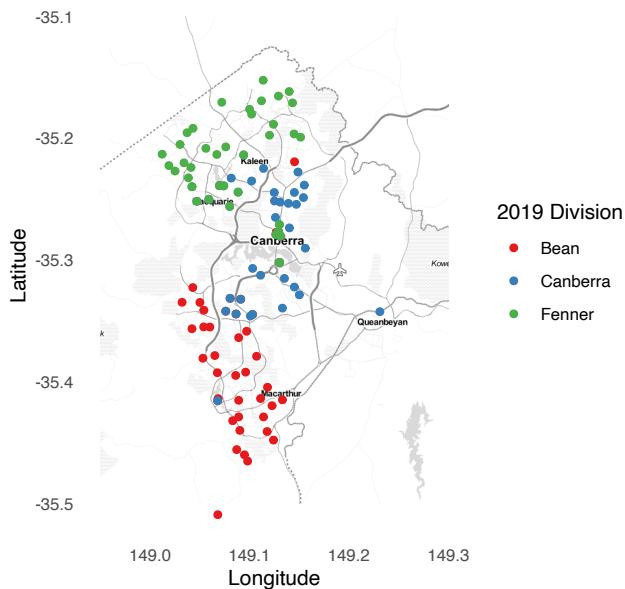
This dataset is for the whole of Australia, but as we are just going to plot the area around Canberra we filter to that and only to booths that are geographic (the AEC has various options for people who are in hospital, or not able to get to a booth, etc, and these are still ‘booths’ in this dataset).

```
Reduce the booths data to only rows with that have latitude and longitude
booths_reduced <-
```

```
booths |>
 filter(State == "ACT") |>
 select(PollingPlaceID, DivisionNm, Latitude, Longitude) |>
 filter(!is.na(Longitude)) |> # Remove rows that do not have a geography
 filter(Longitude < 165) # Remove Norfolk Island
```

Now we can use `ggmap` in the same way as before to plot our underlying tiles, and then build on that using `geom_point()` to add our points of interest.

```
ggmap(canberra_stamen_map,
 extent = "normal",
 maprange = FALSE) +
 geom_point(data = booths_reduced,
 aes(x = Longitude,
 y = Latitude,
 colour = DivisionNm),) +
 scale_color_brewer(name = "2019 Division", palette = "Set1") +
 coord_map(
 projection = "mercator",
 xlim = c(attr(map, "bb")$ll.lon, attr(map, "bb")$ur.lon),
 ylim = c(attr(map, "bb")$ll.lat, attr(map, "bb")$ur.lat)
) +
 labs(x = "Longitude",
 y = "Latitude") +
 theme_minimal() +
 theme(panel.grid.major = element_blank(),
 panel.grid.minor = element_blank())
```



We may like to save the map so that we do not have to draw it every time, and we can do that in the same way as any other graph, using `ggsave()`.

```
ggsave("map.pdf", width = 20, height = 10, units = "cm")
```

Finally, the reason that we used Stamen Maps and OpenStreetMap is because it is open source, but you can also use Google Maps. This requires you to first register a credit card with Google, and specify a key, but with low usage should be free. Using Google Maps, `get_googlemap()`, brings some advantages over `get_stamenmap()`, for instance it will attempt to find a placename, rather than needing to specify a bounding box.

#### 6.4.2 Toronto bike parking

Let us see another example of a static map, this time using Toronto data accessed using `opendatatoronto` (Gelfand, 2020). The dataset that we are going to plot is available here: <https://open.toronto.ca/dataset/street-furniture-bicycle-parking/>.

```
Based on: https://open.toronto.ca/dataset/street-furniture-bicycle-parking/.
library(opendatatoronto)
(The string identifies the package.)
resources <- list_package_resources("71e6c206-96e1-48f1-8f6f-0e804687e3be")
```

```
In this case there is only one dataset within this resource so just need the first one
bike_parking_locations <- filter(resources, row_number()==1) |> get_resource()
```

```
head(bike_parking_locations)
#> # A tibble: 6 x 20
#> `_id` OBJECTID ID ADDRESSNUMBERTEXT ADDRESSSTREET
#> <dbl> <dbl> <chr> <chr> <chr>
#> 1 3613143 4 BP-11699 70 The Pond Rd
#> 2 3613144 9 BP-11900 8 Assiniboine Rd
#> 3 3613145 56 BP-11338 1495 Queen St W
#> 4 3613146 65 BP-03501 8 Kensington Ave
#> 5 3613147 100 BP-03280 87 Avenue Rd
#> 6 3613148 109 BP-12883 21 Canniff St
#> # ... with 15 more variables: FRONTINGSTREET <chr>,
#> # SIDE <chr>, FROMSTREET <chr>, DIRECTION <chr>,
#> # SITEID <lgl>, WARD <chr>, BIA <chr>, ASSETTYPE <chr>,
#> # STATUS <chr>, SDE_STATE_ID <dbl>, X <dbl>, Y <dbl>,
#> # LONGITUDE <dbl>, LATITUDE <dbl>, geometry <chr>
```

First, we need to clean the data are little. There are some clear errors in the ADDRESSNUMBERTEXT field, but not too many, so we just ignore it and focus on the data that we are interested in.

```
bike_data <- tibble(
 ward = bike_parking_locations$WARD,
 id = bike_parking_locations$ID,
 status = bike_parking_locations$STATUS,
 street_address = paste(
 bike_parking_locations$ADDRESSNUMBERTEXT,
 bike_parking_locations$ADDRESSSTREET
),
 latitude = bike_parking_locations$LATITUDE,
 longitude = bike_parking_locations$LONGITUDE
)
rm(bike_parking_locations)
```

Some of the bike racks were temporary so remove them and also let us just look at the area around the University of Toronto, which is Ward 11

```
Only keep ones that still exist
bike_data <-
 bike_data |>
```

```

filter(status == "Existing") |>
select(-status)

bike_data <- bike_data |>
filter(ward == 11) |>
select(-ward)

```

If you look at the dataset at this point, then you will notice that there is a row for every bike parking spot. But we do not really need to know that, because sometimes there are lots right next to each other. Instead, we'd just like the one point. So, we want to create a count by address, and then just get one instance per address.

```

bike_data <-
 bike_data |>
 group_by(street_address) |>
 mutate(number_of_spots = n(),
 running_total = row_number()
) |>
 ungroup() |>
 filter(running_total == 1) |>
 select(-id, -running_total)

head(bike_data)
#> # A tibble: 6 x 4
#> street_address latitude longitude number_of_spots
#> <chr> <dbl> <dbl> <int>
#> 1 8 Kensington Ave 43.7 -79.4 1
#> 2 87 Avenue Rd 43.7 -79.4 4
#> 3 162 Mc Caul St 43.7 -79.4 1
#> 4 147 Baldwin St 43.7 -79.4 2
#> 5 888 Yonge St 43.7 -79.4 1
#> 6 180 Elizabeth St 43.7 -79.4 10

```

Now we can grab our tile and add our bike rack data onto it.

```

bbox <-
c(
 left = -79.420390,
 bottom = 43.642658,
 right = -79.383354,
 top = 43.672557
)

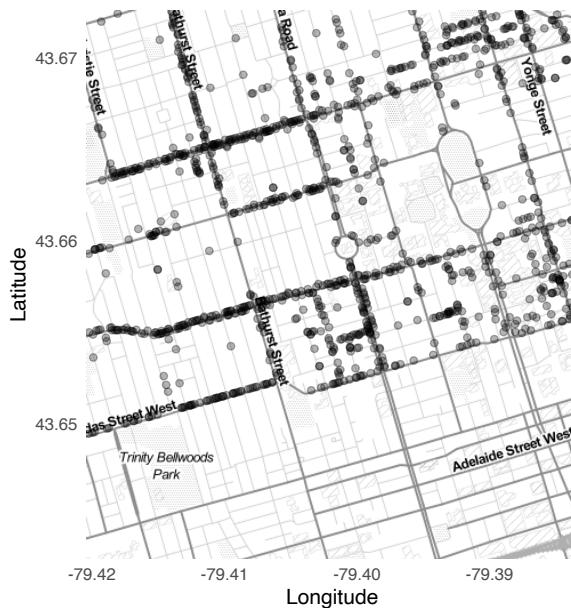
```

```

toronto_stamen_map <-
 get_stamenmap(bbox, zoom = 14, maptype = "toner-lite")

ggmap(toronto_stamen_map, maprange = FALSE) +
 geom_point(data = bike_data,
 aes(x = longitude,
 y = latitude),
 alpha = 0.3) +
 labs(x = "Longitude",
 y = "Latitude") +
 theme_minimal()

```



### 6.4.3 Geocoding

To this point we assumed that we already had geocoded data. But the places ‘Canberra, Australia’, or ‘Ottawa, Canada’, are just names, they do not actually inherently have a location. In order to plot them we need to get a latitude and longitude for them. The process of going from names to coordinates is called geocoding.

There are a range of options to geocode data in R, but `tidygeocoder` is especially useful (Cambon and Belanger, 2021). To get started using the package we need a data frame of locations.

```

some_locations <-
 tibble(city = c('Canberra', 'Ottawa'),
 country = c('Australia', 'Canada'))

tidygeocoder::geo(city = some_locations$city,
 country = some_locations$country,
 method = 'osm')
#> Passing 2 addresses to the Nominatim single address geocoder
#> Query completed in: 2 seconds
#> # A tibble: 2 x 4
#> city country lat long
#> <chr> <chr> <dbl> <dbl>
#> 1 Canberra Australia -35.3 149.
#> 2 Ottawa Canada 45.4 -75.7

```

## 6.5 Exercises and tutorial

### 6.5.1 Exercises

1. I have a dataset that contains measurements of height (in cm) for a sample of 300 penguins, who are either the Adeline or Emperor species. I am interested in visualizing the distribution of heights by species in a graphical way. Please discuss whether a pie chart is an appropriate type of graph to use. What about a box and whisker plot? Finally, what are some considerations if you made a histogram? [Please write a paragraph or two for each aspect.]
2. Assume the dataset and columns exist. Would this code work? data  
|> ggplot(aes(x = col\_one)) |> geom\_point() (pick one)?
  - a. Yes
  - b. No
3. If I have categorical data, which geom should I use to plot it (pick one)?
  - a. geom\_bar()
  - b. geom\_point()
  - c. geom\_abline()
  - d. geom\_boxplot()
4. Why are boxplots often inappropriate (pick one)?
  - a. They hide the full distribution of the data.
  - b. They are hard to make.
  - c. They are ugly.

- d. The mode is clearly displayed.
- 5. Which of the following, if any, are elements of the layered grammar of graphics (Wickham, 2010) (select all that apply)?
  - a. A default dataset and set of mappings from variables to aesthetics.
  - b. One or more layers, with each layer having one geometric object, one statistical transformation, one position adjustment, and optionally, one dataset and set of aesthetic mappings.
  - c. Colors that enable the reader to understand the main point.
  - d. A coordinate system.
  - e. The facet specification.
  - f. One scale for each aesthetic mapping used.

### 6.5.2 Tutorial

Using R Markdown, please create a graph using `ggplot2` and a map using `ggmap` and add explanatory text to accompany both. This should take one or two pages for each of them.

For the graph, please reflect on [Vanderplas et al. \(2020\)](#) and add a few paragraphs about the different options that you considered that the graph more effective.

For the map, please reflect on the following quote from Heather Krause: ‘maps only show people who aren’t invisible to the makers’ as well as Chapter 3 from [D’Ignazio and Klein \(2020\)](#) and add a few paragraphs related to this.



# 7

---

## Interactive communication

---

### Required material

- Read *M-F-E-O: postcards + distill*, (Presmanes Hill, 2021a).
- Read *Building a blog with distill*, (Mock, 2020).
- Read *Geocomputation with R*, Chapter 2 ‘Geographic data in R’, (Lovelace et al., 2019).
- Read *Mastering Shiny*, Chaper 1 ‘Your first Shiny app’, (Wickham, 2021a).

### Key concepts and skills

- Building a website using within the R environment using: `postcards` (Kross, 2021), `distill` (Allaire et al., 2021), and `blogdown` (Xie et al., 2021a).
- Adding interaction to maps, using `leaflet` (Cheng et al., 2021) and `mapdeck` (Cooley, 2020).
- Adding interaction to graphs, using `Shiny` (Chang et al., 2021).

### Key libraries

- `blogdown` (Xie et al., 2021a)
- `distill` (Allaire et al., 2021)
- `leaflet` (Cheng et al., 2021)
- `mapdeck` (Cooley, 2020)
- `postcards` (Kross, 2021)
- `tidyverse` (Wickham et al., 2019a)
- `usethis` (Wickham and Bryan, 2020)

### Key functions

- `blogdown::serve_site()`
- `distill::create_article()`
- `leaflet::addCircleMarkers()`
- `leaflet::addLegend()`
- `leaflet::addMarkers()`
- `leaflet::addTiles()`
- `leaflet::leaflet()`
- `mapdeck:::add_scatterplot()`
- `mapdeck:::mapdeck()`
- `mapdeck:::mapdeck_style()`
- `shiny::fluidPage()`

- `shiny::mainPanel()`
  - `shiny::plotOutput()`
  - `shiny::renderPlot()`
  - `shiny::shinyApp()`
  - `usethis::edit_r_environ()`
  - `usethis::use_git()`
  - `usethis::use_github()`
- 

## 7.1 Introduction

Books and papers have been the primary mediums for communication for thousands of years. But with the rise of computers, and especially the internet, in recent decades, these static approaches have been complemented with interactive approaches. Fundamentally, the internet is about making files available to others. If we additionally allow them to do something with what we make available, then we need to take a variety of additional aspects into consideration.

In this chapter we begin by covering how to create and publish a website. This serves as a place to host a portfolio of work. After that we cover adding interaction to maps and graphs, which are two that nicely lend themselves to this.

---

## 7.2 Making a website

A website is a critical part of communication. For instance, it is a place to make a portfolio of work publicly available. One way to make a website is to use `blogdown` (Xie et al., 2021a). `blogdown` is a package that allows you to make websites, not just blogs, notwithstanding its name, largely within R Studio. It builds on ‘Hugo’, which is a popular general framework for making websites. `blogdown` enables us to freely and quickly get a website up-and-running. It is easy to add content from time-to-time. And it integrates with R Markdown, which makes it easy to share work. But `blogdown` is brittle. Because it is so dependent on Hugo, features that work today may not work tomorrow. Also, owners of Hugo templates can update them at any time, without thought to existing users. `blogdown` is a good option if we know what we are doing, or have a specific use-case, or style, in mind. But two other alternatives are better starting points.

The first is `distill` (Allaire et al., 2021). Again, this is an R package that

wraps around another framework, in this case ‘Distill’. But in contrast to Hugo, Distill is more focused on common needs in data science, and is also only maintained by one group, so it may be considered a more stable choice. That said, the default `distill` site is a little plain looking. As such, following Presmanes Hill (2021a), we will pair it with a third option, `postcards` (Kross, 2021).

The third option, and the one that we will start with, is `postcards` (Kross, 2021). This is a tailored solution that creates simple biographical websites that look great. Having set-up GitHub in R Studio, it is literally possible to have a `postcards` website online in five minutes.

### 7.2.1 Postcards

Begin by installing `postcards`, with `install.packages('postcards')` and then creating a new project for the website (‘File’ -> ‘New Project’ -> ‘New Directory’ -> ‘Postcards Website’). We can then pick a name and location for the project, and select a `postcards` theme. In this case, we can start with ‘trestles’ but this can be changed later. Click the option to ‘Open in new session’ and then create the project.

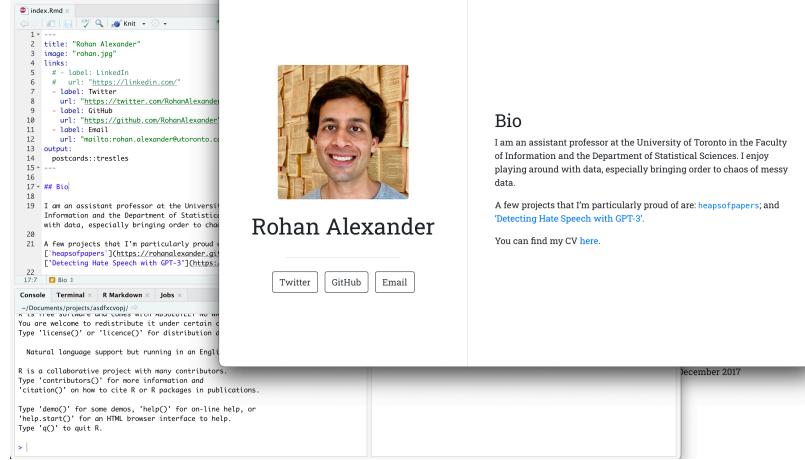
That will open a new file and we can now build the site by clicking ‘Knit’. This will result in a one-page website (Figure 7.1)/

The screenshot shows a one-page website for Frank Hermosillo. At the top left is an RStudio interface showing the code for the website's content. The main content area has a large, centered profile picture of Frank Hermosillo. Below the picture is his name, "Frank Hermosillo", in a large, bold, black font. To the right of the name is a "Bio" section containing a paragraph of text about his research interests and past work. Below the bio are sections for "Education" and "Experience", each with a list of institutions and dates. At the bottom of the page are four social media links: LinkedIn, Twitter, GitHub, and Email. The overall design is clean and professional, typical of the "trestles" theme.

**FIGURE 7.1:** Example of a website made with ‘postcards’ using the ‘trestles’ theme

We can now update the basic content to match our own (Figure 7.2).

Once the details are personalized, then we push it to GitHub. GitHub would



**FIGURE 7.2:** Example of Trestles website with updated details

try to build the site, which we do not want, so we first add a hidden file to turn that off, by running this in the console:

```
file.create('.nojekyll')
```

Then, assuming GitHub was set-up in Chapter 4, we can use `usethis` (Wickham and Bryan, 2020) to get it onto GitHub. We use `use_git()` to initialize a Git repository, and then `use_github()` pushes it to GitHub.

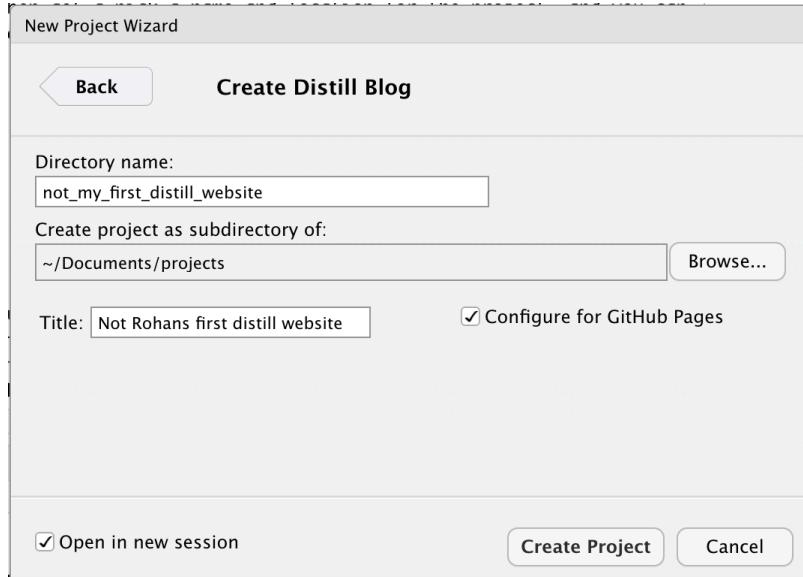
```
library(usethis)
use_git()
use_github()
```

The project will then be on GitHub. We can use GitHub pages to host it: ‘Settings -> Pages’ and then change the source to ‘main’ or ‘master’, depending on your settings. GitHub will let you know the address that you can share to visit your site.

### 7.2.2 Distill

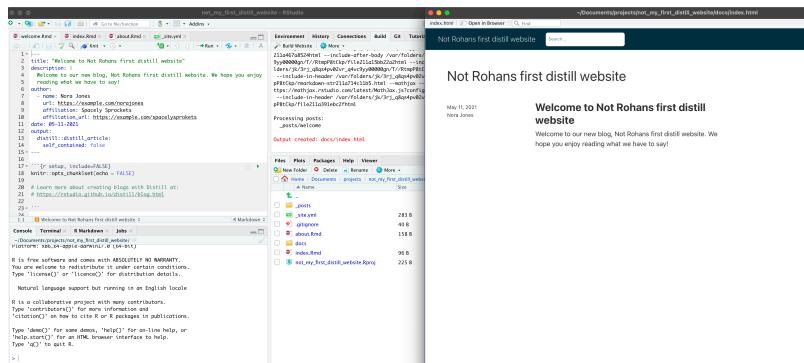
We will now use `distill` (Allaire et al., 2021) to build additional infrastructure around our `postcards` site, following Presmanes Hill (2021a). After that we will explore some of the aspects of `distill` that make it a nice choice, and mention some of the trade-offs. First, we install `distill` with `install.packages('distill')`, and again, create a new project for the website (‘File’ -> ‘New Project’ -> ‘New Directory’ -> ‘Distill Blog’).

We can then pick a name and location for the project, and set a title. Select ‘Configure for GitHub Pages’ and also ‘Open in a new session’. These options can be changed *ex post*. It should look something like Figure 7.3.



**FIGURE 7.3:** Example settings for setting up ‘distill’

At this point we can click ‘Build Website’ in the Build tab, and we should see the default website (Figure 7.4).

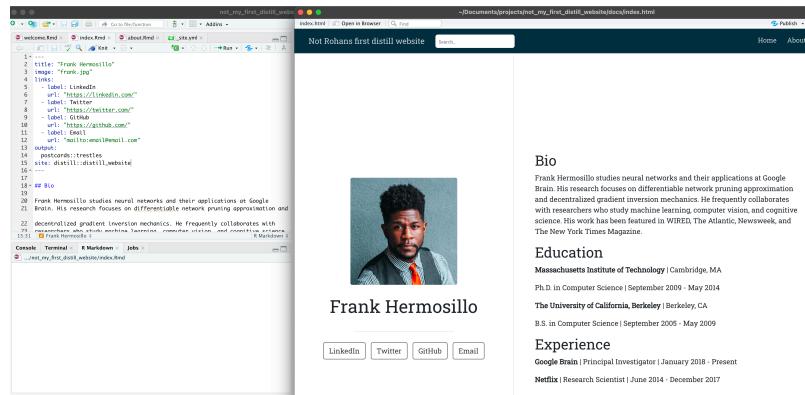


**FIGURE 7.4:** Example of default ‘distill’ website

Again, now we need to update it to reflect our own details. The default for a ‘Distill Blog’ is that the blog is the homepage. We can change that to use the `postcards` page as the homepage. First we change the name of `index.Rmd` to `blog.Rmd` and then create a new ‘trestles’ page:

```
postcards::create_postcard(file = "index.Rmd", template = "trestles")
```

The trestles page will open, and we need to add the following line in the yaml file: `site: distill::distill_website`. In Figure 7.5 it was added at line 16, and then we can rebuild the website.



**FIGURE 7.5:** Updating the yaml to change the homepage

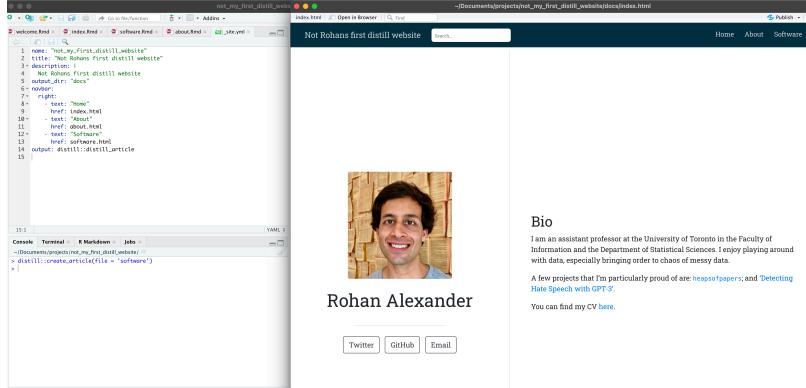
We can make the same changes to the default content as earlier, updating the links, image, and bio. The advantage of using `distill` is that we now have additional pages, not just a one-page website, and we also have a blog. By default, we have an ‘about’ page, but some other pages that may be useful, depending on the particular use-case, include: ‘research’, ‘teaching’, ‘talks’, ‘projects’, ‘software’, ‘datasets’. As an example, we will add and edit a page called ‘software’ using `distill::create_article(file = 'software')`.

That will create and open an R Markdown document. To add it to the website, open `_site.yml` and then add a line to the ‘navbar’ (Figure 7.6). After this is done then we can rebuild the website, and the ‘software’ page will have been added.

We can continue with this process until we are happy with the website. For instance, we may want to add a blog. To do this we follow the same pattern as before, but with ‘blog’ instead of ‘software’.

When we are happy with our website, we can push it to GitHub and then use GitHub Pages to host it, in the same way that we did with the `postcards` site.

Using the `distill` is a good option when we need a multi-page website, but still want a fairly controlled environment. There are many options that can be changed, and Presmanes Hill (2021a) is a good starting point, in addition to the `distill` homepage: <https://rstudio.github.io/distill/>.



**FIGURE 7.6:** Adding another page to the website

That said, `distill` is opinionated. While it is a great option, if we want something a little more flexible then `blogdown` might be a better option.

### 7.2.3 Blogdown

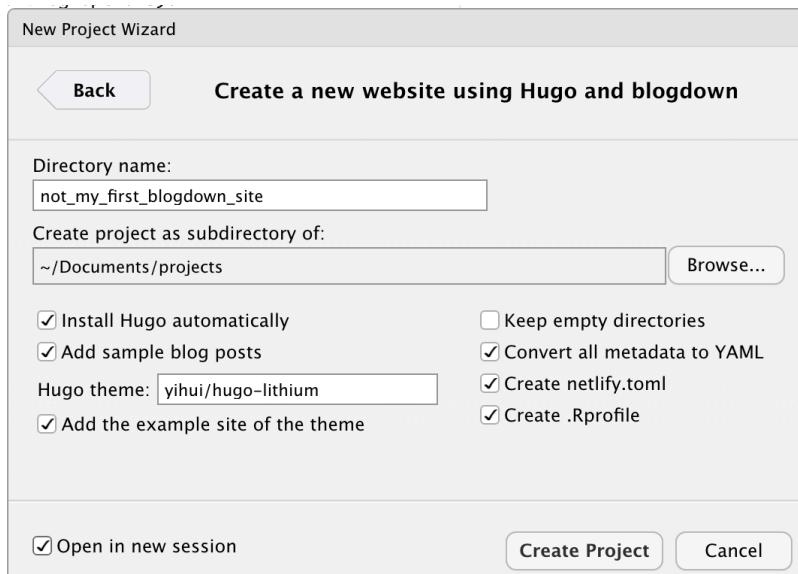
Using `blogdown` (Xie et al., 2021a) is more work than Google sites or Squarespace. It requires a little more knowledge than using a basic Wordpress site. And if you want to customize absolutely every aspect of your website, or need everything to be ‘just so’ then `blogdown` may not be a good option. Further, `blogdown` is still under active development and various aspects may break in future releases. However, `blogdown` allows a variety and level of expression that is not possible with `distill`. Presmanes Hill (2021b) and Xie et al. (2021b) are excellent options for learning more about `blogdown`.

First we will need to install `blogdown` `install.packages("blogdown")`. And then create a new project for the website ('File' -> 'New Project' -> 'New Directory' -> 'Website using blogdown'). At this point you can set a name and location, and also select 'Open in a new session' (Figure 7.7).

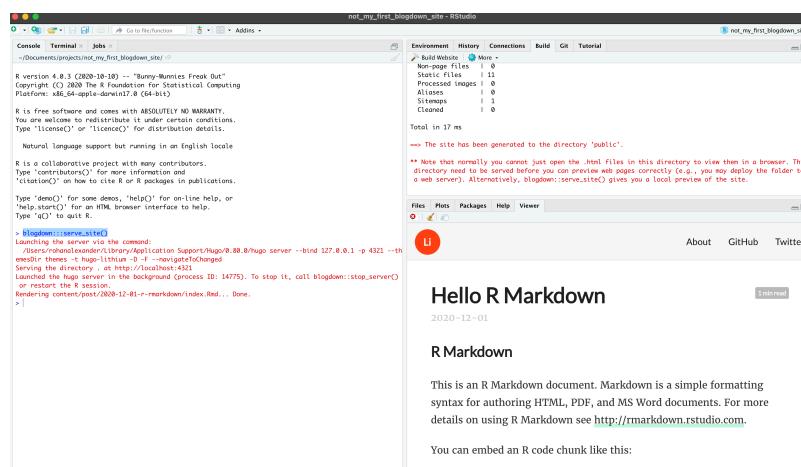
We can click ‘Build Website’ from the ‘Build’ pane, but then an extra step is needed; we need to serve the site `blogdown:::serve_site()`. After this, the site will show in the ‘Viewer’ pane (Figure 7.8).

At this point, the default website is being ‘served’ locally. This means that changes we make will be reflected in the website that we see in the Viewer pane. To see the website in a web browser, click ‘Show in new window’ on the top left of the Viewer. That will open the website using the address that the R Studio also tells you.

We now want to update the content, starting with the ‘About’ section. To do that go to ‘content -> about.md’ and add your own content. One nice aspect



**FIGURE 7.7:** Example settings for setting up blogdown



**FIGURE 7.8:** Serving default blogdown site

of `blogdown` is that it will automatically re-load the content when you save, so you should see your changes immediately show up. You may also like to change the logo. You could do this by adding a square image to ‘public/images/’ and then changing the call to ‘logo.png’ in ‘config.yaml’. When we are happy with it, we can make our website public in the same way as we did for `postcards`.

One advantage of using `blogdown` is that it allows us to use Hugo templates. This provides a large number of beautifully crafted websites. To pick a theme we go to the Hugo themes page: <https://themes.gohugo.io>. There are hundreds of different themes. In general, most of them can be made to work with `blogdown`, but sometimes it can be a bit of a hassle to get them working.

One nice option is Apéro: <https://hugo-apero-docs.netlify.app>. We can specify the use of this theme as part of creating a new site ('File -> New Project -> New Directory -> Website using blogdown'). At this point, in addition to setting the name and location, we can specify a theme. Specifically, in 'Hugo theme' field, we specify a GitHub username and repository, which in this case is 'hugo-apero/apero' (Figure 7.9).

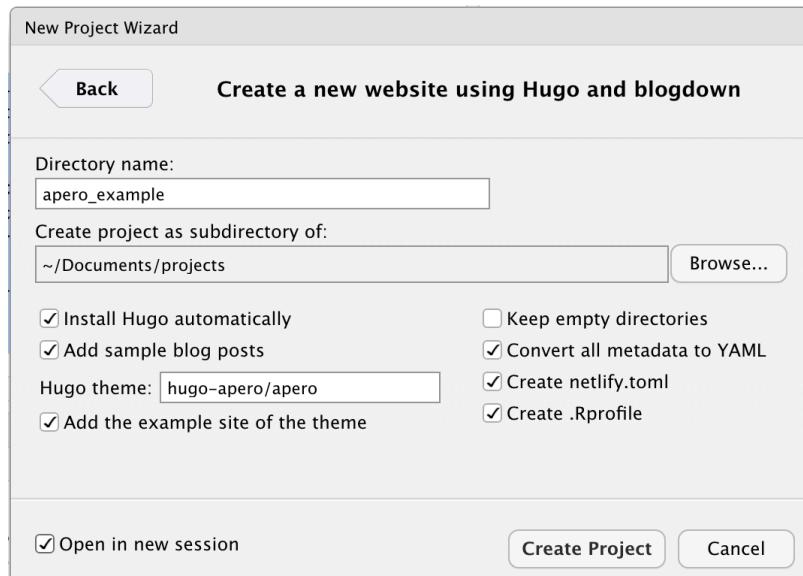


FIGURE 7.9: Using the Apéro theme

### 7.3 Interactive maps

The nice thing about interactive maps is that we can let our user decide what they are interested in. For instance, in the case of Canadian politics, some people will be interested in Toronto ridings, while others will be interested in Manitoba, etc. But it would be difficult to present a map that focuses on both of those, so an interactive map is a great option for allowing users to focus on what they want.

That said, it is important to be cognizant of what we are doing when we build maps, and more broadly, what is being done at scale to enable us to be able to build our own maps. For instance, with regard to Google, McQuire (2019) says:

---

Google began life in 1998 as a company famously dedicated to organising the vast amounts of data on the Internet. But over the last two decades its ambitions have changed in a crucial way. Extracting data such as words and numbers from the physical world is now merely a stepping-stone towards apprehending and organizing the physical world as data. Perhaps this shift is not surprising at a moment when it has become possible to comprehend human identity as a form of (genetic) ‘code’. However, apprehending and organizing the world as data under current settings is likely to take us well beyond Heidegger’s ‘standing reserve’ in which modern technology enframed ‘nature’ as productive resource. In the 21st century, it is the stuff of human life itself—from genetics to bodily appearances, mobility, gestures, speech, and behaviour—that is being progressively rendered as productive resource that can not only be harvested continuously but subject to modulation over time.

---

Does this mean that we should not use or build interactive maps? Of course not. But it is important to be aware of the fact that this is a frontier, and the boundaries of appropriate use are still being determined. Indeed, the literal boundaries of the maps themselves are being consistently determined and updated. The move to digital maps, compared with physical printed maps, means that it is actually possible for different users to be presented with different realities. For instance, ‘...Google routinely takes sides in border disputes. Take, for instance, the representation of the border between Ukraine

and Russia. In Russia, the Crimean Peninsula is represented with a hard-line border as Russian-controlled, whereas Ukrainians and others see a dotted-line border. The strategically important peninsula is claimed by both nations and was violently seized by Russia in 2014, one of many skirmishes over control’ Bensinger (2020).

### 7.3.1 Leaflet

We can use `leaflet` (Cheng et al., 2021) to make interactive maps. The essentials are similar to `ggmap` (Kahle and Wickham, 2013), but there are many additional aspects beyond that. We can redo the bike map from Chapter 6.

In the same way as a graph in `ggplot2` begins with `ggplot()`, a map in `leaflet` begins with `leaflet()`. Here we can specify data, and other options such as width and height. After this, we add ‘layers’ in the same way that we added them in `ggplot2`. The first layer that we add is a tile, using `addTiles()`. In this case, the default is from OpenStreetMap. After that we add markers with `addMarkers()` to show the location of each bike parking spot.

```
library(leaflet)
library(tidyverse)

bike_data <- read_csv("outputs/data/bikes.csv")

leaflet(data = bike_data) |>
 addTiles() # Add default OpenStreetMap map tiles
 addMarkers(lng = bike_data$longitude,
 lat = bike_data$latitude,
 popup = bike_data$street_address,
 label = ~as.character(bike_data$number_of_spots))
```



There are two new arguments. The first is ‘popup’, which is what happens when the user clicks on the marker. Here the address will be provided. The second is ‘label’, which is what happens when the user hovers on the marker. Here the number of parking spots is specified.

We can try another example, this time of the fire stations in Toronto. We can use data from Open Data Toronto using `opendatatoronto` (Gelfand, 2020).

```
library(opendatatoronto)
Get starter code from: https://open.toronto.ca/dataset/fire-station-locations/
fire_stations_locations <- get_resource('9d1b7352-32ce-4af2-8681-595ce9e47b6e')
Grab the lat and long - thanks https://stackoverflow.com/questions/47661354/converting-geometry-
fire_stations_locations <-
 fire_stations_locations |>
 tidyrr::extract(geometry, c('lon', 'lat'), '\\\\((.*), (.*)\\\\)', convert = TRUE)
```

```
head(fire_stations_locations)
#> # A tibble: 6 x 26
#> `_id` ID NAME ADDRESS ADDRESS_POINT_ID ADDRESS_ID
#> <dbl> <dbl> <chr> <chr> <dbl> <dbl>
#> 1 1 21 FIRE ST~ 900 TAPS~ 4236992 363382
#> 2 2 60 FIRE ST~ 106 ASCO~ 764237 70190
#> 3 3 61 FIRE ST~ 65 HENDR~ 819425 127148
#> 4 4 55 FIRE ST~ 260 ADEL~ 12763904 484214
```

```
#> 5 5 24 FIRE ST~ 745 MEAD~ 6349868 357277
#> 6 6 74 FIRE ST~ 140 LANS~ 10757599 157562
#> # ... with 20 more variables: CENTRELINE_ID <dbl>,
#> # MAINT_STAGE <chr>, ADDRESS_NUMBER <dbl>,
#> # LINEAR_NAME_FULL <chr>, POSTAL_CODE <chr>,
#> # GENERAL_USE <chr>, CLASS_FAMILY_DESC <chr>,
#> # ADDRESS_ID_LINK <dbl>, PLACE_NAME <chr>, X <lgl>,
#> # Y <lgl>, LATITUDE <lgl>, LONGITUDE <lgl>,
#> # WARD_NAME <chr>, MUNICIPALITY_NAME <chr>, ...
```

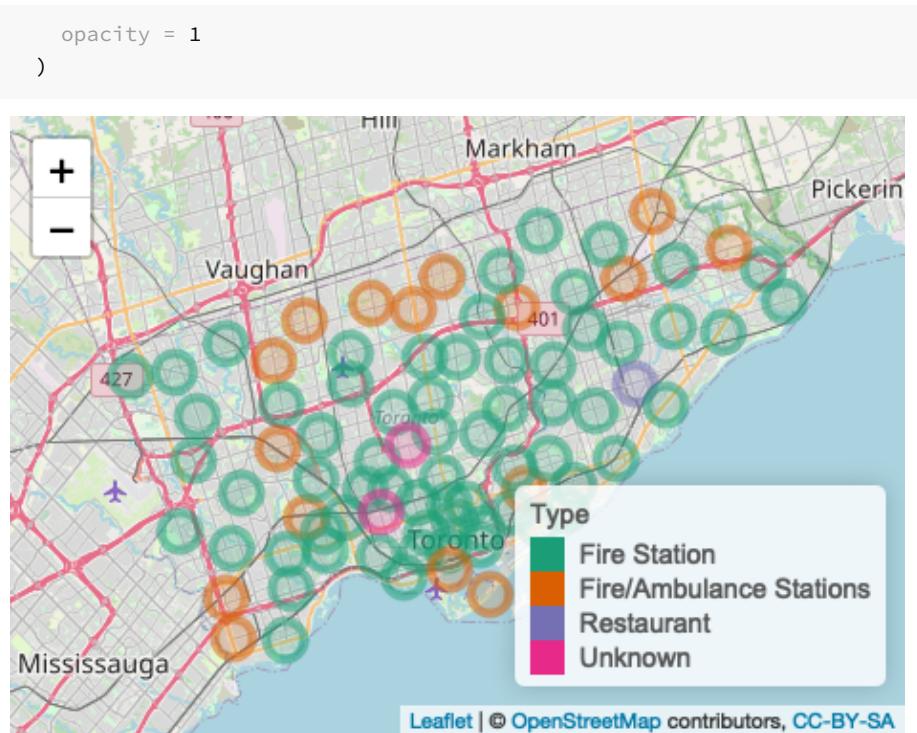
There is a lot of information here, but we will just plot the location of each fire station along with their name and address.

We will introduce a different type of marker here, which is circles. This will allow us to use different colors for the outcomes of each type. There are four possible outcomes: “Fire/Ambulance Stations”, “Fire Station”, “Restaurant”, “Unknown”.

```
library(leaflet)

pal <-
 colorFactor("Dark2", domain = fire_stations_locations$GENERAL_USE |> unique())

leaflet() |>
 addTiles() |> # Add default OpenStreetMap map tiles
 addCircleMarkers(
 data = fire_stations_locations,
 lng = fire_stations_locations$lon,
 lat = fire_stations_locations$lat,
 color = pal(fire_stations_locations$GENERAL_USE),
 popup = paste(
 "Name:",
 as.character(fire_stations_locations$NAME),
 "
",
 "Address:",
 as.character(fire_stations_locations$ADDRESS),
 "
"
)
) |>
 addLegend(
 "bottomright",
 pal = pal,
 values = fire_stations_locations$GENERAL_USE |> unique(),
 title = "Type",
```



### 7.3.2 Mapdeck

`mapdeck` (Cooley, 2020) is based on WebGL. This means the web browser will do a lot of work for us. This enables us to accomplish things with `mapdeck` that `leaflet` struggles with, such as larger datasets.

To this point we have used ‘stamen maps’ as our underlying tile, but `mapdeck` uses ‘Mapbox’: <https://www.mapbox.com/>. This requires registering an account and obtaining a token. This is free and only needs to be done once. Once we have that token we add it to our R environment (the details of this process are covered in Chapter 8) by running `usethis::edit_r_environ()`, which will open a text file. There we can add our Mapbox secret token.

```
MAPBOX_TOKEN = 'PUT_YOUR_MAPBOX_SECRET_HERE'
```

We then save this ‘.Renvironment’ file, and restart R (‘Session’ -> ‘Restart R’).

Having obtained a token, we can create a plot of our firefighters data from earlier.

```
library(mapdeck)
#>
#> Attaching package: 'mapdeck'
#> The following object is masked from 'package:tibble':
#>
#> add_column

mapdeck(style = mapdeck_style('dark')
) |>
 add_scatterplot(
 data = fire_stations_locations,
 lat = "lat",
 lon = "lon",
 layer_id = 'scatter_layer',
 radius = 10,
 radius_min_pixels = 5,
 radius_max_pixels = 100,
 tooltip = "ADDRESS"
)
#> Registered S3 method overwritten by 'jsonify':
#> method from
#> print.json jsonlite
```



## 7.4 Shiny

`shiny` (Chang et al., 2021) is a way of making interactive web applications using R. It is fun, but fiddly. Here we are going to step through one way to take advantage of `shiny`. Which is to quickly add some interactivity to our graphs. We will return to `shiny` in more detail in Chapter 20.

We are going to make an interactive graph based on the ‘babynames’ dataset from `babynames` (Wickham, 2019b). First, we will build a static version (Figure 7.10).

```
library(babynames)
library(tidyverse)

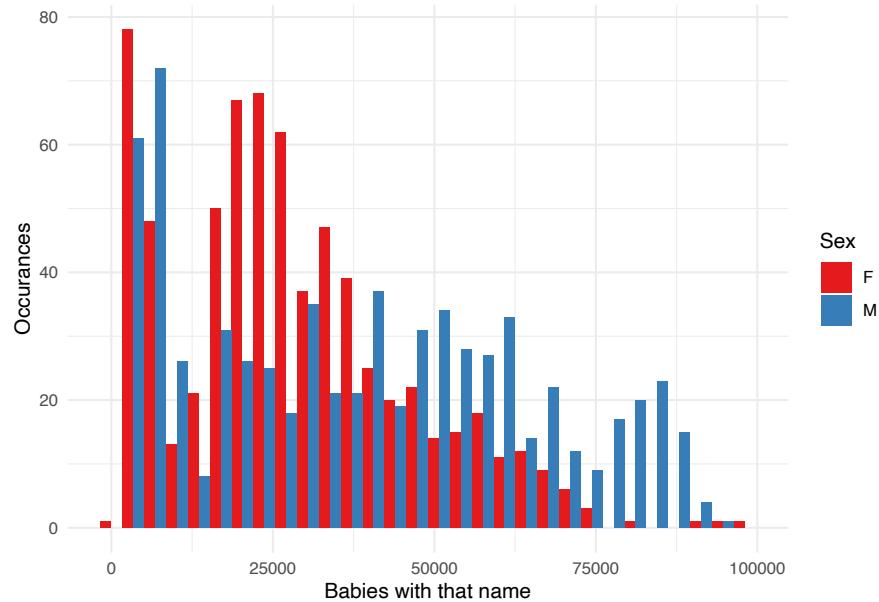
top_five_names_by_year <-
 babynames |>
 group_by(year, sex) |>
 arrange(desc(n)) |>
 slice_head(n = 5)

top_five_names_by_year |>
 ggplot(aes(x = n, fill = sex)) +
 geom_histogram(position = "dodge") +
 theme_minimal() +
 scale_fill_brewer(palette = "Set1") +
 labs(x = "Babies with that name",
 y = "Occurrences",
 fill = "Sex")
```

We can see the most popular boys names tend to be more clustered, compared with the most-popular girls names, which may be more spread out. However, one thing that we might be interested in is how the effect of the ‘bins’ parameter shapes what we see. We might like to use interactivity to explore different values.

To get started, create a new `shiny` app (‘File -> New File -> Shiny Web App’). Give it a name, such as ‘not\_my\_first\_shiny’ and then leave all the other options as the default. A new file ‘app.R’ will open and we click ‘Run app’ to see what it looks like.

Now replace the content in that file, ‘app.R’, with the content below, and then again click ‘Run app’



**FIGURE 7.10:** Popular baby names

```

library(shiny)

Define UI for application that draws a histogram
ui <- fluidPage(
 # Application title
 titlePanel("Count of names for five most popular names each year.",

 # Sidebar with a slider input for number of bins
 sidebarLayout(sideBarPanel(
 sliderInput(
 inputId = "number_of_bins",
 label = "Number of bins:",
 min = 1,
 max = 50,
 value = 30
)
),

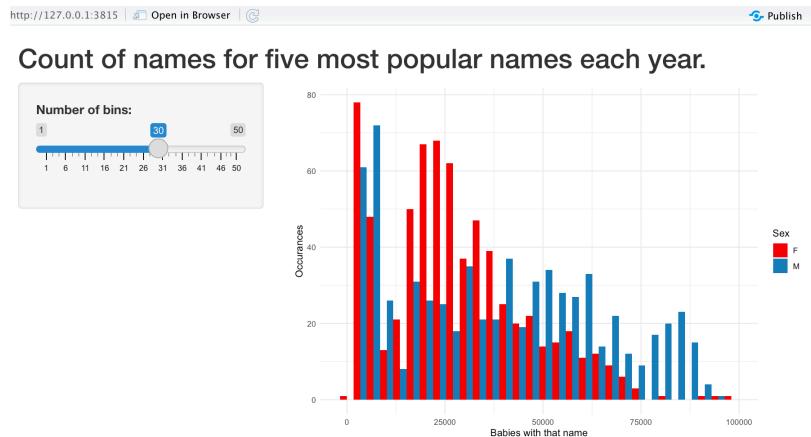
 # Show a plot of the generated distribution
 mainPanel(plotOutput("distPlot")))
)

```

```
Define server logic required to draw a histogram
server <- function(input, output) {
 output$distPlot <- renderPlot({
 # Draw the histogram with the specified number of bins
 top_five_names_by_year |>
 ggplot(aes(x = n, fill = sex)) +
 geom_histogram(position = "dodge", bins = input$number_of_bins) +
 theme_minimal() +
 scale_fill_brewer(palette = "Set1") +
 labs(x = "Babies with that name",
 y = "Occurrences",
 fill = "Sex")
 })
}

Run the application
shinyApp(ui = ui, server = server)
```

You should find that you are served an interactive graph where you can change the number of bins and it should look like Figure 7.11.



**FIGURE 7.11:** Example of Shiny app where the user controls the number of bins

---

## 7.5 Exercises and tutorial

### 7.5.1 Exercises

### 7.5.2 Tutorial

### 7.5.3 Paper

At about this point, Paper 2 (Appendix B.2) would be appropriate.

---

---

## Part III

# Measure and acquire



# 8

---

## Gather data

---

**STATUS:** Under construction.

### Recommended reading

- Algorithmic thinking in the public interest: navigating technical, legal, and ethical hurdles to web scraping in the social sciences
- Benoit, Kenneth, 2019, ‘Text as data: An overview’, 17 July, <https://kenbenoit.net/pdfs/28%20Benoit%20Text%20as%20Data%20draft%202.pdf>.
- Bolton, Liza, 2019, ‘A quick look at museums per capita’, 26 March, <http://blog.dataembassy.co.nz/museums-per-capita/>.
- Bryan, Jennifer, and Jim Hester, 2020, *What They Forgot to Teach You About R*, Chapter 7, <https://rstats.wtf/index.html>.
- Cardoso, Tom, 2019, ‘Introduction to scraping’, <https://github.com/tomcardoso/intro-to-scraping>.
- Clavelle, Tyler, 2017, ‘Using R to extract data from web APIs’, 5 June, <https://www.tylerclavelle.com/code/2017/randapis/>.
- Cooksey, Brian, 2014, ‘An Introduction to APIs’, Zapier, 22 April, <https://zapier.com/learn/apis/>.
- Dogucu, Mine, and Mine Çetinkaya-Runde, 2020 , ‘Web Scraping in the Statistics and Data Science Curriculum: Challenges and Opportunities’, 6 May.
- Gelfand, Sharla, 2019, ‘Crying @ Sephora’, 8 November, <https://sharla.party/post/crying-sephora/>.
- Goldman, Shayna, 2019, ‘How Much Do NHL Players Really Make? Part 2: Taxes’, <https://hockey-graphs.com/2019/01/08/how-much-do-nhl-players-really-make-part-2-taxes/>.
- Graham, Shawn, 2019, ‘Scraping with rvest’, 7 November, <https://electricarchaeology.ca/2019/11/07/scraping-with-rvest/>.
- Henze, Martin, 2020, ‘Web Scraping with rvest + Astro Throwback’, 23 January, <https://heads0rtails.github.io/2020/01/23/rvest-intro-astro/>.

- Hudon, Caitlin, 2017, ‘Blue Christmas: A data-driven search for the most depressing Christmas song’, 22 December, <https://caitlinhudon.com/2017/12/22/blue-christmas/>.
- Luscombe, Alex, 2020, ‘A Gentle Introduction to Tesseract OCR’, 3 June, <https://alexluscombe.ca/post/ocr-tutorial/>.
- Luscombe, Alex, 2020, ‘Getting your .pdfs into R’, 5 August, <https://alexluscombe.ca/post/r-pdf-tools/>.
- Luscombe, Alex, 2020, ‘Parsing your .pdfs in R’, 10 August, <https://alexluscombe.ca/post/parsing-pdfs/>.
- Marshall, James, ‘HTML Made Really Easy’, <https://www.jmarshall.com/easy/html/>.
- Marshall, James, ‘HTTP Made Really Easy’, <https://www.jmarshall.com/easy/http/>.
- Nakagawara, Ryo, 2020, ‘Intro to {polite} Web Scraping of Soccer Data with R!’, 14 May, <https://ryo-n7.github.io/2020-05-14-webscrape-soccer-data-with-R/>.
- Pavlik, Kaylin, 2020, ‘How do fiber types appear together in yarn blends?’, 17 February, <https://www.kaylinpavlik.com/ravelry-yarn-fibers/>.
- Silge, Julia and David Robinson, 2020, *Text Mining with R*, Chapters 1, 3, and 6, <https://www.tidytextmining.com/>.
- Silge, Julia, 2017, ‘Scraping CRAN with rvest’, 5 March, <https://juliasilge.com/blog/scraping-cran/>.
- Smale, David, 2020, ‘Daniel Johnston’, <https://davidsmale.netlify.com/portfolio/daniel-johnston/>.
- Taddy, Matt, 2019, *Business Data Science*, Chapter 8, pp. 231-259.
- Wickham, Hadley, ‘Managing Secrets’, <https://cran.r-project.org/web/packages/httr/vignettes/secrets.html>.
- Wickham, Hadley, 2014, ‘rvest: easy web scraping with R’, 24 November, <https://blog.rstudio.com/2014/11/24/rvest-easy-web-scraping-with-r/>.
- Wickham, Hadley, nd, ‘Getting started with httr’, <https://cran.r-project.org/web/packages/httr/vignettes/quickstart.html>.

### Recommended viewing

- D’Agostino McGowan, Lucy, 2020 ‘Harnessing the Power of the Web via R Clients for Web APIs’, talk at ASA Joint Statistical Meeting 2018, [https://www.lucymcgowan.com/talk/asa\\_joint\\_statistical\\_meeting\\_2018/](https://www.lucymcgowan.com/talk/asa_joint_statistical_meeting_2018/).
- Tatman, Rachel, 2018, ‘Character Encoding and You’, 21 February, <https://youtu.be/2U9EHYqc59Y>.

### Key concepts/skills/etc

- Use APIs where possible because the data provider has specified the data they would like to make available to you, and the conditions under which they are making it available.
- Often R packages have been written to make it easier to use APIs.
- Use R environments to manage your keys.
- Using the verb GET ('a GET request') means providing a URL and the server will return something, using the verb POST ('a POST request') means providing some data and the server will deal with that data.
- Cleaning data
- Graphing data to tell a story
- Respectfully scraping data
- Approaching extracting text from PDFs as a workflow.
- Planning what is needed at the start.
- Starting small and then iterating.
- Putting in place checks.
- Gathering text data.
- Preparing text datasets.

### Key libraries

- `babynames`
- `broom`
- `dplyr`
- `ggplot2`
- `gutenbergr`
- `janitor`
- `jsonlite`
- `pdftools`
- `purrr`
- `rtweet`
- `rvest`
- `spotifyr`
- `stringi`
- `tidymodels`
- `tidytext`
- `tidyverse`
- `usethis`

### Key functions/etc

- `as_factor()`
- `as_tibble()`
- `bind_tf_idf()`
- `c()`
- `case_when()`

- `cat()`
  - `edit_r_environ()`
  - `file()`
  - `fromJSON()`
  - `function()`
  - `GET()`
  - `get_artist_audio_features()`
  - `get_favorites()`
  - `get_my_top_artists_or_tracks()`
  - `html_node()`
  - `html_nodes()`
  - `html_text()`
  - `pdf_data()`
  - `pdf_text()`
  - `pmap_dfr()`
  - `read_html()`
  - `readRDS()`
  - `safely()`
  - `search_tweets()`
  - `sleep()`
  - `tesseract()`
  - `unnest_tokens()`
  - `walk2()`
  - `write_html()`
  - `write_lines()`
- 

## 8.1 APIs

[Get some interesting ones from here: [https://bookdown.org/paul/apis\\_for\\_social\\_scientists/](https://bookdown.org/paul/apis_for_social_scientists/)]

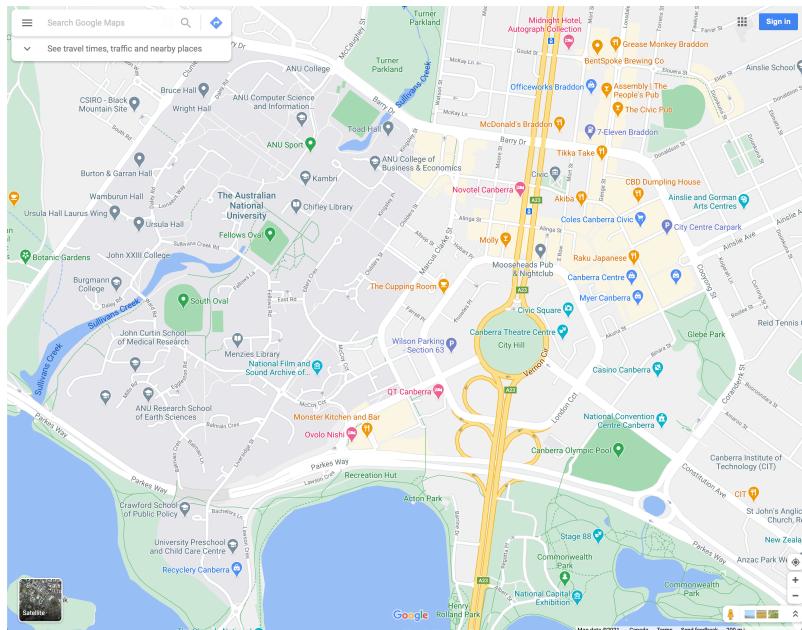
In everyday language, and for our purposes, an Application Programming Interface (API) is simply a situation in which someone has set up specific files on their computer such that you can follow their instructions to get them. For instance, when you use a gif on Slack, Slack asks Giphy's server for the appropriate gif, Giphy's server gives that gif to Slack and then Slack inserts it into your chat. The way in which Slack and Giphy interact is determined by Giphy's API. More strictly, an API is just an application that runs on a server that we access using the HTTP protocol.

In our case, we are going to focus on using APIs for gathering data. I'll tailor the language that I use toward that:

[a]n API is the tool that makes a website's data digestible for a computer. Through it, a computer can view and edit data, just like a person can by loading pages and submitting forms.

Cooksey (2014), Chapter 1.

For instance, you could go to Google Maps<sup>1</sup> and then scroll and click and drag to center the map on Canberra, Australia, or you could just paste this into your browser: <https://www.google.ca/maps/@-35.2812958,149.1248113,16z>. You just used the Google Maps API.<sup>2</sup> The result should be a map that looks something like Figure 8.1 .



**FIGURE 8.1:** Example of Google Maps, as at 25 January 2021.

The advantage of using an API is that the data provider specifies exactly the data that they are willing to provide, and the terms under which they will

<sup>1</sup><https://www.google.ca/maps>

<sup>2</sup>There are at least six great coffee shops shown just in this section of map including: Mocan & Green Grout; The Cupping Room; Barrio Collective Coffee; Lonsdale Street Cafe; Two Before Ten; and Red Brick. There are also two coffee shops that I love but that most wouldn't classify as 'great' including: The Street Theatre Cafe; and the CBE Cafe.

provide it. These terms may include things like rate limits (i.e. how often you can ask for data), and what you can do with the data (e.g. maybe you're not allowed to use it for commercial purposes, or to republish it, or whatever). Additionally, because the API is being provided specifically for you to use it, it is less likely to be subject to unexpected changes. Because of this it is ethically and legally clear that when an API is available you should try to use it.

We're going to run through some case studies interacting with APIs in R. In the first we will deal directly with an API. That works and is a handy skill to have, but there are a lot of R packages that wrap around APIs making it easier for you to use an API within 'familiar surroundings'. I'll also run through two fun APIs that have R packages built around them.

## 8.2 Case study - arXiv

In this section we introduce GET requests in which we use an API directly. We will use the `httr` package (Wickham, 2019c). A GET request tries to obtain some specific data and the main argument is `url`. Exactly as before with the Google Maps example! In that case, the specific information was a map and some information about it.

For this example we'll look at arXiv<sup>3</sup>, which is a repository for academic articles before they go through peer-review. I'll ask arXiv to return some information about a paper that I recently uploaded with a former student. The content that is returned will be a series of information about that paper.

```
install.packages('httr')
library(httr)
arxiv <- httr::GET('http://export.arxiv.org/api/query?id_list=2101.05225')
class(arxiv)
#> [1] "response"
content(arxiv, "text") %>%
 cat("\n")
#> <?xml version="1.0" encoding="UTF-8"?>
#> <feed xmlns="http://www.w3.org/2005/Atom">
#> <link href="http://arxiv.org/api/query?search_query%3D%26id_list%3D2101.05225%26start%3D0%26max_results%3D100" type="self" rel="root"/>
#> <title type="html">ArXiv Query: search_query=&id_list=2101.05225&start=0&max_results=100</title>
#> <id>http://arxiv.org/api/p9UZyl2Vt0cHwPSKinDSThE23qI</id>
#> <updated>2022-01-26T00:00:00-05:00</updated>
#> <opensearch:totalResults xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">1</opensearch:totalResults>
```

<sup>3</sup><https://arxiv.org>

```
#> <opensearch:startIndex xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">0</opensearch:
#> <opensearch:itemsPerPage xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">10</opensear
#> <entry>
#> <id>http://arxiv.org/abs/2101.05225v1</id>
#> <updated>2021-01-13T17:37:07Z</updated>
#> <published>2021-01-13T17:37:07Z</published>
#> <title>On consistency scores in text data with an implementation in R</title>
#> <summary> In this paper, we introduce a reproducible cleaning process for the text
#> extracted from PDFs using n-gram models. Our approach compares the originally
#> extracted text with the text generated from, or expected by, these models using
#> earlier text as stimulus. To guide this process, we introduce the notion of a
#> consistency score, which refers to the proportion of text that is expected by
#> the model. This is used to monitor changes during the cleaning process, and
#> across different corpuses. We illustrate our process on text from the book Jane
#> Eyre and introduce both a Shiny application and an R package to make our
#> process easier for others to adopt.
#> </summary>
#> <author>
#> <name>Ke-Li Chiu</name>
#> </author>
#> <author>
#> <name>Rohan Alexander</name>
#> </author>
#> <arxiv:comment xmlns:arxiv="http://arxiv.org/schemas/atom">13 pages, 0 figures</arxiv:comm
#> <link href="http://arxiv.org/abs/2101.05225v1" rel="alternate" type="text/html"/>
#> <link title="pdf" href="http://arxiv.org/pdf/2101.05225v1" rel="related" type="application/
#> <arxiv:primary_category xmlns:arxiv="http://arxiv.org/schemas/atom" term="cs.CL" scheme="ht
#> <category term="cs.CL" scheme="http://arxiv.org/schemas/atom"/>
#> </entry>
#> </feed>
#>
```

We get a variety of information about this paper including the title, abstract, and authors.

---

### 8.3 Case study - rtweet

Twitter is a rich source of text and other data. The Twitter API is the way in which Twitter ask that you interact with Twitter in order to gather these data. The `rtweet` package (Kearney, 2019) is built around this API and allows

us to interact with it in ways that are similar to using any other R package. Initially all you need is a regular Twitter account.

Get started by installing the library if you need and then calling it.

```
install.packages('rtweet')
library(rtweet)
library(tidyverse)
```

To get started we need to authorise rtweet. We start that process by calling a function from the package.

```
getFavorites(user = "RohanAlexander")
```

This will open a browser on your computer, and you will then have to log into your regular Twitter account as shown in Figure 8.2.

Once that is done we can actually get my favourites and then save them.

```
rohans_favs <- getFavorites("RohanAlexander")

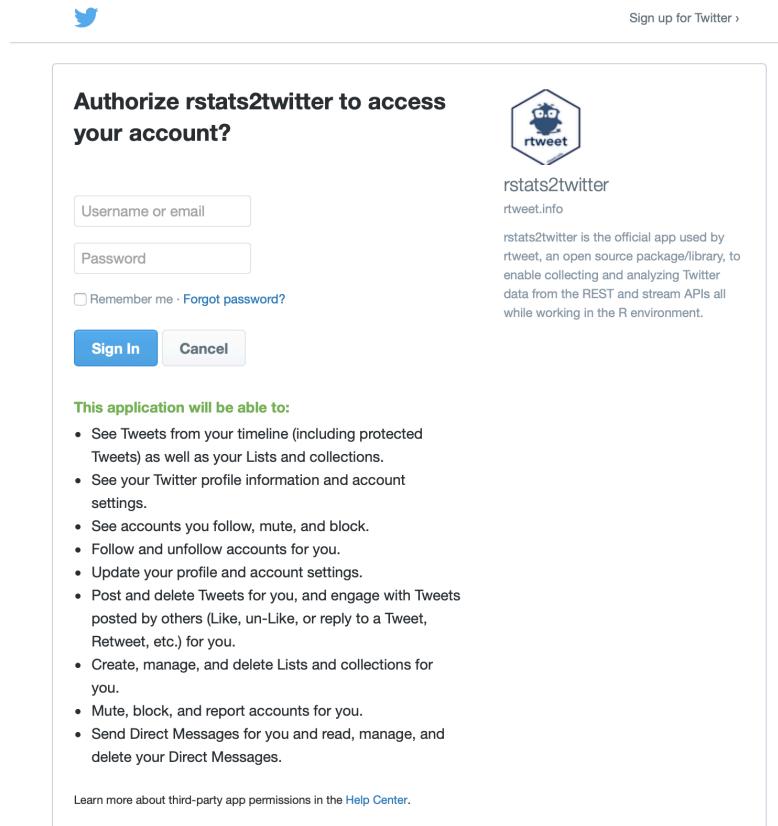
saveRDS(rohans_favs, "dont_push/rohans_favs.rds")
```

And then looking at the most recent favourite, we can see it was when Professor Bolton tweeted about one of the stellar students in ISSC.

```
rohans_favs %>%
 arrange(desc(created_at)) %>%
 slice(1) %>%
 select(screen_name, text)
#> # A tibble: 1 x 2
#> screen_name text
#> <chr> <chr>
#> 1 les_ja I've signed an offer letter, so I think I can formally announce: ~
```

Let's look at who is tweeting about R, using one of the common R hashtags: #rstats. I've removed retweets so that we hopefully get some actual interesting projects.

```
rstats_tweets <- search_tweets(
 q = "#rstats",
 include_rts = FALSE
)
```

**FIGURE 8.2:** rtweet authorisation page

```
saveRDS(rstats_tweets, "dont_push/rstats_tweets.rds")
```

And then have a look at them.

```
names(rstats_tweets)
#> [1] "user_id" "status_id"
#> [3] "created_at" "screen_name"
#> [5] "text" "source"
#> [7] "display_text_width" "reply_to_status_id"
#> [9] "reply_to_user_id" "reply_to_screen_name"
#> [11] "is_quote" "is_retweet"
#> [13] "favorite_count" "retweet_count"
#> [15] "quote_count" "reply_count"
```

```

#> [17] "hashtags" "symbols"
#> [19] "urls_url" "urls_t.co"
#> [21] "urls_expanded_url" "media_url"
#> [23] "media_t.co" "media_expanded_url"
#> [25] "media_type" "ext_media_url"
#> [27] "ext_media_t.co" "ext_media_expanded_url"
#> [29] "ext_media_type" "mentions_user_id"
#> [31] "mentions_screen_name" "lang"
#> [33] "quoted_status_id" "quoted_text"
#> [35] "quoted_created_at" "quoted_source"
#> [37] "quoted_favorite_count" "quoted_retweet_count"
#> [39] "quoted_user_id" "quoted_screen_name"
#> [41] "quoted_name" "quoted_followers_count"
#> [43] "quoted_friends_count" "quoted_statuses_count"
#> [45] "quoted_location" "quoted_description"
#> [47] "quoted_verified" "retweet_status_id"
#> [49] "retweet_text" "retweet_created_at"
#> [51] "retweet_source" "retweet_favorite_count"
#> [53] "retweet_retweet_count" "retweet_user_id"
#> [55] "retweet_screen_name" "retweet_name"
#> [57] "retweet_followers_count" "retweet_friends_count"
#> [59] "retweet_statuses_count" "retweet_location"
#> [61] "retweet_description" "retweet_verified"
#> [63] "place_url" "place_name"
#> [65] "place_full_name" "place_type"
#> [67] "country" "country_code"
#> [69] "geo_coords" "coords_coords"
#> [71] "bbox_coords" "status_url"
#> [73] "name" "location"
#> [75] "description" "url"
#> [77] "protected" "followers_count"
#> [79] "friends_count" "listed_count"
#> [81] "statuses_count" "favourites_count"
#> [83] "account_created_at" "verified"
#> [85] "profile_url" "profile_expanded_url"
#> [87] "account_lang" "profile_banner_url"
#> [89] "profile_background_url" "profile_image_url"

rstats_tweets %>%
 select(screen_name, text) %>%
 head()
#> # A tibble: 6 x 2
#> screen_name text
#> <chr> <chr>

```

```
#> 1 RahaPhD "#WFH multitasking woes: I was just sitting here, working on ~
#> 2 AmandaKMontoya "Teaching with the #PublishingPaidMe data this week in my intr~
#> 3 digitalke1 "130 #MachineLearning ProjectsSolved and Explained\n@ruben_arc~
#> 4 digitalke1 "#Infographic: 6 simple steps to effectively analyse data.\nVi~
#> 5 dataclaudius "When Did the US Senate Best Reflect the US Population? via #r~
#> 6 alexpghayes "has anyone written an #rstats package to interface with SNAP ~
```

There is a bunch of other things that you can do just using a regular user account, and if you're interested then you should try the examples in the `rtweet` package documentation: <https://rtweet.info/index.html>. But more is available once you register as a developer (<https://developer.twitter.com/en/apply-for-access>). The Twitter API document is surprisingly readable, and you may enjoy some of it: <https://developer.twitter.com/en/docs>.

When I introduced APIs I said that the ‘data provider specifies exactly the data that they are willing to provide...’ and we have certainly been able to take advantage of what they provide. But I continued ‘...and the terms under which they will provide it’ and here we haven’t done our part. In particular, I took some tweets and saved them. If I had pushed these to GitHub, then it’s possible I may have accidentally stored sensitive information if there happened to be some in the tweets. Or if I had taken enough tweets to start to do some reasonable statistical analysis then even if there wasn’t sensitive information, I may have violated the terms if I had pushed those saved tweets to GitHub. Finally, I linked a Twitter username, in this case @Liza\_Bolton with Professor Bolton. I happened to ask her if this was okay, but if I hadn’t done that then I would have been violating the Twitter terms of service.

If you use Twitter data, please take a moment to look at the terms: <https://developer.twitter.com/en/developer-terms/more-on-restricted-use-cases>.

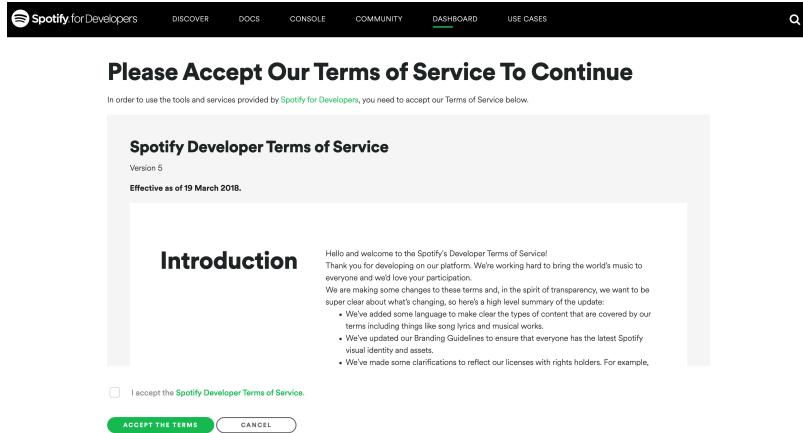
## 8.4 Case study - *spotifyr*

For the next example I will introduce the `spotifyr` package (Thompson et al., 2020). Again, this is a wrapper that has been developed around an API, in this case the Spotify API. You should install the package from the developer’s GitHub repo using `devtools` (Wickham et al., 2020b).

```
devtools::install_github('charlie86/spotifyr')
library(spotifyr)
```

In order to use this account, you need a Spotify Developer Account, which you can set-up here: <https://developer.spotify.com/dashboard/>. That’ll have

you log in with your Spotify details and then accept their terms (it's worth looking at some of these and I'll follow up on a few below) as in Figure 8.3.



**FIGURE 8.3:** rtweet authorisation page

What we need from here is a ‘Client ID’ and you can just fill out some basic details. In our case we probably ‘don’t know’ what we’re building, which means that Spotify requires us to use a non-commercial agreement, which is fine. In order to use the Spotify API we need a Client ID and a Client Secret.

These are things that you want to keep to yourself. There are a variety of ways of keeping this secret, (and my understanding is that a helpful package is on its way) but we’ll keep them in our System Environment. In this way, when we push to GitHub they won’t be included. To do this we need to be careful about the naming, because `spotifyr` will look in our environment for specifically named keys.

To do this we are going to use the `usethis` package (Wickham and Bryan, 2020). If you don’t have that then please install it. There is a file called ‘`Renviron`’ which we will open and add our secrets to. This file also controls things like your default library location and more information is available at Lopp (2017) and Bryan and Hester (2020).

```
usethis::edit_r_environ()
```

When you run that function it will open a file. There you can add your Spotify secrets.

```
SPOTIFY_CLIENT_ID = 'PUT_YOUR_CLIENT_ID_HERE'
SPOTIFY_CLIENT_SECRET = 'PUT_YOUR_SECRET_HERE'
```

Save your ‘Renvironment’ file, and then restart R (Session -> Restart R). You can now draw on that variable when you need.

Some functions that require your secrets as arguments will now just work. For instance, we will get information about Radiohead using `get_artist_audio_features()`. One of the arguments is `authorization`, but as that is set to default to look at the R Environment, we don’t need to do anything further.

```
radiohead <- get_artist_audio_features('radiohead')
saveRDS(radiohead, "inputs/radiohead.rds")
```

```
radiohead <- readRDS("inputs/radiohead.rds")

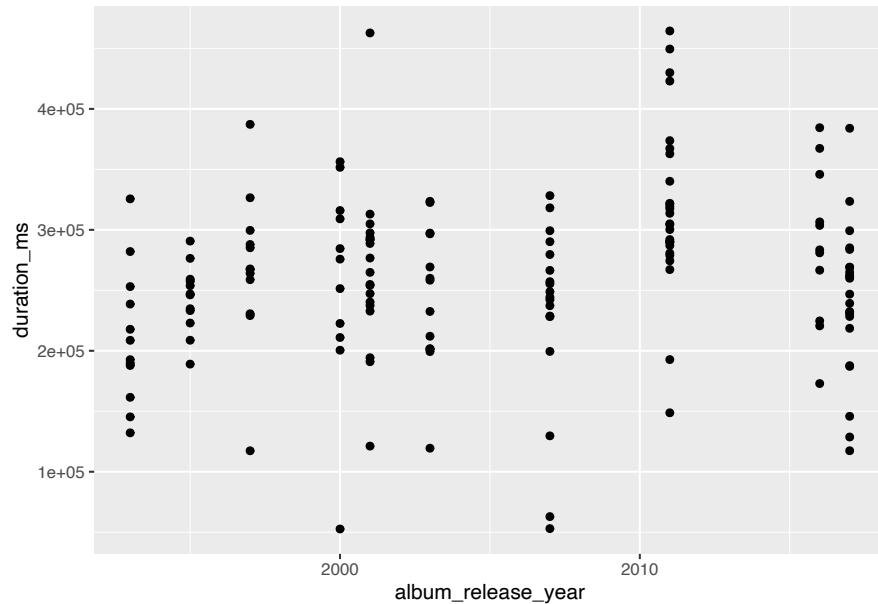
names(radiohead)
#> [1] "artist_name" "artist_id"
#> [3] "album_id" "album_type"
#> [5] "album_images" "album_release_date"
#> [7] "album_release_year" "album_release_date_precision"
#> [9] "danceability" "energy"
#> [11] "key" "loudness"
#> [13] "mode" "speechiness"
#> [15] "acousticness" "instrumentalness"
#> [17] "liveness" "valence"
#> [19] "tempo" "track_id"
#> [21] "analysis_url" "time_signature"
#> [23] "artists" "available_markets"
#> [25] "disc_number" "duration_ms"
#> [27] "explicit" "track_href"
#> [29] "is_local" "track_name"
#> [31] "track_preview_url" "track_number"
#> [33] "type" "track_uri"
#> [35] "external_urls.spotify" "album_name"
#> [37] "key_name" "mode_name"
#> [39] "key_mode"

radiohead %>%
 select(artist_name, track_name, album_name) %>%
 head()
#> artist_name track_name
#> 1 Radiohead Airbag - Remastered
#> 2 Radiohead Paranoid Android - Remastered
#> 3 Radiohead Subterranean Homesick Alien - Remastered
#> 4 Radiohead Exit Music (For a Film) - Remastered
```

```
#> 5 Radiohead Let Down - Remastered
#> 6 Radiohead Karma Police - Remastered
#>
#> album_name
#> 1 OK Computer OKNOTOK 1997 2017
#> 2 OK Computer OKNOTOK 1997 2017
#> 3 OK Computer OKNOTOK 1997 2017
#> 4 OK Computer OKNOTOK 1997 2017
#> 5 OK Computer OKNOTOK 1997 2017
#> 6 OK Computer OKNOTOK 1997 2017
```

Let's just make a quick graph looking at track length over time.

```
radiohead %>%
 ggplot(aes(x = album_release_year, y = duration_ms)) +
 geom_point()
```



Just because we can, let's settle an argument. I've always said that Radiohead are quite depressing, but they're my wife's favourite band. Let's see how depressing they are. Spotify provides various information about each track, including 'valence', which Spotify define<sup>4</sup> as '(a) measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more

<sup>4</sup><https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/>

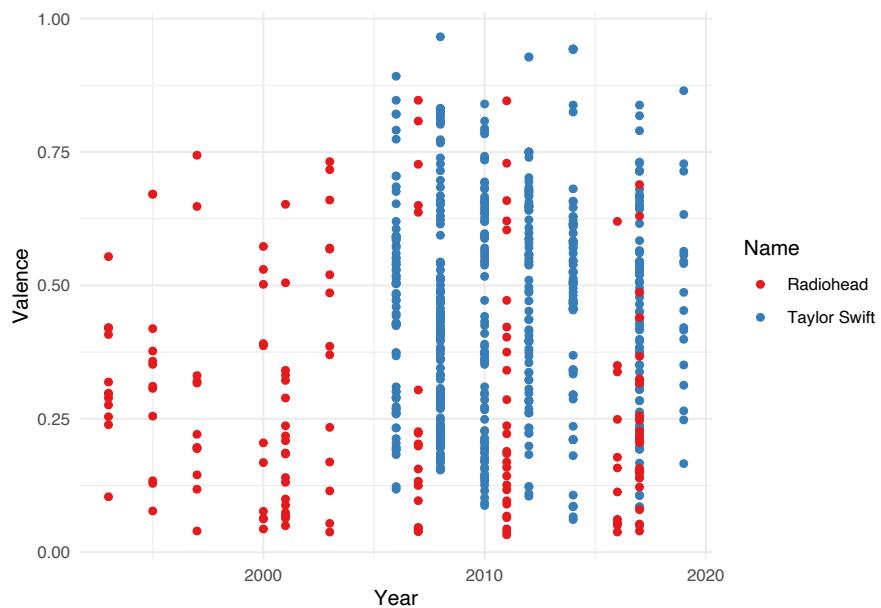
positive (e.g. happy, cheerful, euphoric), while tracks with low valence sound more negative (e.g. sad, depressed, angry).<sup>7</sup> Higher values are happier. Let's compare someone who we know it likely to be happy - Taylor Swift - with Radiohead.

```
swifty <- get_artist_audio_features('taylor swift')
saveRDS(swifty, "inputs/swifty.rds")
```

```
swifty <- readRDS("inputs/swifty.rds")

tibble(name = c(swifty$artist_name, radiohead$artist_name),
 year = c(swifty$album_release_year, radiohead$album_release_year),
 valence = c(swifty$valence, radiohead$valence)
) %>%
ggplot(aes(x = year, y = valence, color = name)) +
 geom_point() +
 theme_minimal() +
 labs(x = "Year",
 y = "Valence",
 color = "Name") +
 scale_color_brewer(palette = "Set1")
```



Finally, for the sake of embarrassment, let's look at our most played artists.

```
top_artists <- get_my_top_artists_or_tracks(type = 'artists', time_range = 'long_term', limit = 20

saveRDS(top_artists, "inputs/top_artists.rds")

top_artists <- readRDS("inputs/top_artists.rds")

top_artists %>%
 select(name, popularity)
#> name popularity
#> 1 Radiohead 81
#> 2 Bombay Bicycle Club 66
#> 3 Drake 100
#> 4 Glass Animals 74
#> 5 JAY-Z 85
#> 6 Laura Marling 65
#> 7 Sufjan Stevens 75
#> 8 Vampire Weekend 73
#> 9 Sturgill Simpson 65
#> 10 Nick Drake 66
#> 11 Dire Straits 78
#> 12 Lorde 80
#> 13 Marian Hill 65
#> 14 José González 68
#> 15 Stevie Wonder 79
#> 16 Disclosure 82
#> 17 Ben Folds Five 52
#> 18 Ainslie Wills 40
#> 19 Coldplay 89
#> 20 alt-J 75
```

So pretty much my wife and I like what everyone else likes, with the exception of Ainslie Wills, who is an Australian and I suspect we used to listen to her when we were homesick.

How amazing that we live in a world that all that information is available with very little effort or cost.

Again, there is a lot more at the package's website: <https://www.rcharlie.com/spotifyr/>. A very nice little application of the Spotify API using some statistical analysis is [Pavlik \(2019\)](#).

## 8.5 Scraping

### 8.5.1 Introduction

Web-scraping is a way to get data from websites into R. Rather than going to a website ourselves through a browser, we write code that does it for us. This opens up a lot of data to us, but on the other hand, it is not typically data that is being made available for these purposes and so it is important to be respectful of it. While generally not illegal, the specifics with regard to the legality of web-scraping depends on jurisdictions and the specifics of what you're doing, and so it is also important to be mindful of this. And finally, web-scraping imposes a cost on the website host, and so it is important to reduce this to the extent that it's possible.

That all said, web-scraping is an invaluable source of data. But they are typically datasets that can be created as a by-product of someone trying to achieve another aim. For instance, a retailer may have a website with their products and their prices. That has not been created deliberately as a source of data, but we can scrape it to create a dataset. As such, the following principles guide my web-scraping.

1. Avoid it. Try to use an API wherever possible.
2. Abide by their desires. Some websites have a file ‘robots.txt’ that contains information about what they are comfortable with scrapers doing, for instance ‘<https://www.google.com/robots.txt>’. If they have one of these then you should read it and abide by it.
3. Reduce the impact.
  - Firstly, slow down your scraper, for instance, rather than having it visit the website every second, slow it down (using `sys.sleep()`). If you’re only after a few hundred files then why not just have it visit once a minute, running in the background overnight?
  - Secondly, consider the timing of when you run the scraper. For instance, if it’s a retailer then why not set your script to run from 10pm through to the morning, when fewer customers are likely to need the site? If it’s a government website and they have a big monthly release then why not avoid that day?
4. Take only what you need. For instance, don’t scrape the entire of Wikipedia if all you need is to know the names of the 10 largest cities in Canada. This reduces the impact on their website and allows you to more easily justify what you are doing.
5. Only scrape once. Save everything as you go so that you don’t have to re-collect data. Similarly, once you have the data, you should keep that separate and not modify it. Of course, if you need data

over time then you will need to go back, but this is different to needlessly re-scraping a page.

6. Don't republish the pages that you scraped. (This is in contrast to datasets that you create from it.)
7. Take ownership and ask permission if possible. At a minimum level your scripts should have your contact details in them. Depending on the circumstances, it may be worthwhile asking for permission before you scrape.

### 8.5.2 Getting started

Web-scraping is possible by taking advantage of the underlying structure of a webpage. We use patterns in the HTML/CSS to get the data that we want. To look at the underlying HTML/CSS you can either: 1) open a browser, right-click, and choose something like 'Inspect'; or 2) save the website and then open it with a text editor rather than a browser.

HTML/CSS is a markup language comprised of matching tags. If you want text to be bold then you would use something like:

```
My bold text
```

Similarly, if you want a list then you start and end the list as well as each item.

```

 Learn webscraping
 Do data science
 Profit

```

When scraping we will search for these tags.

To get started, this is some HTML/CSS from my website. Let's say that we want to grab my name from it. We can see that the name is in bold, so we want to probably focus on that feature and extract it.

```
website_extract <- "<p>Hi, I'm Rohan Alexander.</p>"
```

We will use the `rvest` package [Wickham \(2019d\)](#).

```
install.packages("rvest")
library(rvest)
```

```
rohans_data <- read_html(website_extract)

rohans_data
#> {html_document}
#> <html>
#> [1] <body><p>Hi, I'm Rohan Alexander.</p></body>
```

The language used by `rvest` to look for tags is ‘node’, so we will focus on bold nodes. By default `html_nodes()` returns the tags as well. We can focus on the text that they contain, using `html_text()`.

```
rohans_data %>%
 html_nodes("b")
#> {xml_nodeset (1)}
#> [1] Rohan

first_name <-
 rohans_data %>%
 html_nodes("b") %>%
 html_text()

first_name
#> [1] "Rohan"
```

The result is that we learn my first name.

## 8.6 Case study - Rohan's books

### 8.6.1 Introduction

In this case study we are going to scrape a list of books that I own, clean it, and look at the distribution of the first letters of author surnames. It is slightly more complicated than the example above, but the underlying approach is the same - download the website, look for the nodes of interest, extract the information, clean it.

### 8.6.2 Gather

Again, the key library that we are using is the `rvest` library. This makes it easier to download a website, and to then navigate the html to find the aspects that we are interested in. You should create a new project in a new folder (File

-> New Project). Within that new folder you should make three new folders: `inputs`, `outputs`, and `scripts`.

In the scripts folder you should write and save a script along these lines. This script loads the libraries that we need, then visits my website, and saves a local copy.

```
Contact details
Title: Get data from rohanalexander.com
Purpose: This script gets data from Rohan's website about the books that he
owns. It calls his website and then saves the dataset to inputs.
Author: Rohan Alexander
Contact: rohan.alexander@utoronto.ca
Last updated: 20 May 2020

Set up workspace
library(rvest)
library(tidyverse)

Get html
rohans_data <- read_html("https://rohanalexander.com/bookshelf.html")
This takes a website as an input and will read it into R, in the same way that we
can read a, say, CSV into R.

write_html(rohans_data, "inputs/my_website/raw_data.html")
Always save your raw dataset as soon as you get it so that you have a record
of it. This is the equivalent of, say, write_csv() that we have used earlier.
```

### 8.6.3 Clean

Now we need to navigate the HTML to get the aspects that we want, and to then put them into some sensible structure. I always try to get the data into a tibble as early as possible. While it's possible to work with the nested data, I move to a tibble so that the usual verbs that I'm used to can be used.

In the scripts folder you should write and save a new R script along these lines. First, we need to add the top matter, read in the libraries and the data that we scraped.

```
Contact details
Title: Clean data from rohanalexander.com
Purpose: This script cleans data that was downloaded in 01-get_data.R.
```

```

Author: Rohan Alexander
Contact: rohan.alexander@utoronto.ca
Pre-requisites: Need to have run 01_get_data.R and have saved the data.
Last updated: 20 May 2020

Set up workspace
library(tidyverse)
library(rvest)

rohans_data <- read_html("inputs/my_website/raw_data.html")

rohans_data
#> {html_document}
#> <html xmlns="http://www.w3.org/1999/xhtml" lang="" xml:lang="">
#> [1] <head>\n<meta http-equiv="Content-Type" content="text/html; charset=UTF-8 ...
#> [2] <body>\n\n<!--radix_placeholder_front_matter-->\n\n<script id="distill-fr ...

```

Now we need to identify the data that we are interested in using html tags and convert it to a tibble. If you look at the website, then you should notice that we are likely trying to focus on list items (Figure 8.4).

Let's look at the source (Figure 8.5).

There's a lot of debris, but scrolling down we eventually get to a list (Figure 8.6).

The tag for a list item is 'li', so we modify the earlier code to focus on that and to get the text.

```

Clean data
Identify the lines that have books on them based on the list html tag
text_data <- rohans_data %>%
 html_nodes("li") %>%
 html_text()

all_books <- tibble(books = text_data)

head(all_books)
#> # A tibble: 6 x 1
#> books
#> <chr>
#> 1 "-“A Little Life”, Hanya Yanighara. Recommended by Lauren."
#> 2 "“The Andromeda Strain”, Michael Crichton."
#> 3 "“Is There Life After Housework”, Don Aslett.\nGot given this at the Museum o~

```

## Bookshelf

Inspired by Patrick Collison's [version](#), this is an incomplete (so far) and unordered list of the books that I own. I've been a bit more liberal than Patrick and included books that I've read and would like to own (designated by "-"). If you have recommendations, then please [get in touch](#). I usually only buy books that I like, but bolded books are ones I especially like.

- -"A Little Life", Hanya Yanigihara. Recommended by Lauren.
- "The Andromeda Strain", Michael Crichton.
- "Is There Life After Housework", Don Aslett.  
Got given this at the Museum of Clean in Pocatello, Idaho. The museum was surprisingly good!
- "The Chosen", Chaim Potok.
- "The Forsyth Saga", John Galsworthy.
- "Freakonomics", Steven Levitt and Stephen Dubner.
- "Tess of the D'Urbervilles" Thomas Hardy.
- "The Da Vinci Code", Dan Brown.
- "King Charles III", Anthony Holden.
- "The Road Washes Out in Spring", Baron Wormser.  
Stayed at the author's house, so bought some of his books and particularly liked this one about how his family lived off the grid in New England. Liked it even better after living in Amherst for a while.
- "The Terminal Man", Michael Crichton.
- "The Shepherd's Hut", Tim Winton.  
Present from Helen.
- "**Code: The Hidden Language of Computer Hardware and Software**",  
**Charles Petzold**.
- "Hitch-22", Christopher Hitchens.
- "Audacity of Hope", Barack Obama.
- "Kennedy", Ted Sorenson.
- "The Son Also Rises", Greg Clark.

**FIGURE 8.4:** Some of Rohan's books

```

1 <!DOCTYPE html>
2
3 <html xmlns="http://www.w3.org/1999/xhtml" lang="" xml:lang="">
4
5 <head>
6 <meta charset="utf-8"/>
7 <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
8 <meta http-equiv="X-UA-Compatible" content="IE=Edge,chrome=1"/>
9 <meta name="generator" content="distill" />
10
11 <style type="text/css">
12 /* Hide doc at startup (prevent jankiness while JS renders/transitions) */
13 body {
14 visibility: hidden;
15 }
16 </style>
17
18 <!--radix_placeholder_import_source-->
19 <!--/radix_placeholder_import_source-->
20

```

**FIGURE 8.5:** Source code for top of the page

```

1343
1344 <div class="d-article">
1345
1346 “A Little Life”, Hanya Yanagihara. Recommended by Lauren.
1347 “The Andromeda Strain”, Michael Crichton.
1348 “Is There Life After Housework”, Don Aslett.

1349 Got given this at the Museum of Clean in Pocatello, Idaho. The mu
1350 “The Chosen”, Chaim Potok.
1351 “The Forsyth Saga”, John Galsworthy.
1352 “Freakonomics”, Steven Levitt and Stephen Dubner.
1353 “Tess of the D’Urbervilles”, Thomas Hardy.
1354 “The Da Vinci Code”, Dan Brown.
1355 “King Charles III”, Anthony Holden.
1356 “The Road Washes Out in Spring”, Baron Wormser.

1357 Stayed at the author’s house, so bought some of his books and par
1358 “The Terminal Man”, Michael Crichton.
1359 “The Shepherd’s Hut”, Tim Winton.

1360 Present from Helen.
1361 “Code: The Hidden Language of Computer Hardware and S
1362 “Hitch-22”, Christopher Hitchens.

```

**FIGURE 8.6:** Source code for list

```

#> 4 ““The Chosen”, Chaim Potok.”
#> 5 ““The Forsyth Saga”, John Galsworthy.”
#> 6 ““Freakonomics”, Steven Levitt and Stephen Dubner.”

```

We now need to clean the data. First we want to separate the title and the author

```

All content is just one string, so need to separate title and author
all_books <-
 all_books %>%
 separate(books, into = c("title", "author"), sep = "")
```

```
Remove leading comma and clean up the titles a little
all_books <-
 all_books %>%
 mutate(author = str_remove_all(author, ", "),
 author = str_squish(author),
 title = str_remove(title, "“"),
 title = str_remove(title, "^-"))
)

head(all_books)
#> # A tibble: 6 x 2
#> title author
#> <chr> <chr>
#> 1 A Little Life Hanya Yanighara. Recommended by Lauren.
#> 2 The Andromeda Strain Michael Crichton.
#> 3 Is There Life After Housework Don Aslett. Got given this at the Museum of Cle-
#> 4 The Chosen Chaim Potok.
#> 5 The Forsyth Saga John Galsworthy.
#> 6 Freakonomics Steven Levitt and Stephen Dubner.
```

Finally, some specific cleaning is needed.

```
Some authors have comments after their name, so need to get rid of them, although there are some
J. K. Rowling.
M. Mitchell Waldrop.
David A. Price
all_books <-
 all_books %>%
 mutate(author = str_replace_all(author,
 c("J. K. Rowling." = "J K Rowling.",
 "M. Mitchell Waldrop." = "M Mitchell Waldrop.",
 "David A. Price" = "David A Price")))
)

) %>%
separate(author, into = c("author_correct", "throw_away"), sep = "\\.", extra = "drop") %>%
select(-throw_away) %>%
rename(author = author_correct)

Some books have multiple authors, so need to separate them
One has multiple authors:
"Daniela Witten, Gareth James, Robert Tibshirani, and Trevor Hastie"
all_books <-
 all_books %>%
```

```

mutate(author = str_replace(author,
 "Daniela Witten, Gareth James, Robert Tibshirani, and Trevor Hastie",
 "Daniela Witten and Gareth James and Robert Tibshirani and Trevor Hastie"),
 separate(author, into = c("author_first", "author_second", "author_third", "author_fourth"), sep = ","),
 pivot_longer(cols = starts_with("author_"),
 names_to = "author_position",
 values_to = "author") %>%
 select(-author_position) %>%
 filter(!is.na(author))

head(all_books)
#> # A tibble: 6 x 2
#> title author
#> <chr> <chr>
#> 1 A Little Life Hanya Yanighara
#> 2 The Andromeda Strain Michael Crichton
#> 3 Is There Life After Housework Don Aslett
#> 4 The Chosen Chaim Potok
#> 5 The Forsyth Saga John Galsworthy
#> 6 Freakonomics Steven Levitt

```

It looks there is some at the end because I have a best of. I'll just get rid of those manually because it's not the focus.

```

all_books <-
 all_books %>%
 slice(1:118)

```

#### 8.6.4 Explore

Finally, just because we have the data now, so we may as well try to do something with it, let's look at the distribution of the first letter of the author names.

```

all_books %>%
 mutate(
 first_letter = str_sub(author, 1, 1)
) %>%
 group_by(first_letter) %>%
 count()
#> # A tibble: 21 x 2
#> # Groups: first_letter [21]
#> first_letter n

```

```
#> <chr> <int>
#> 1 "" 1
#> 2 "A" 8
#> 3 "B" 5
#> 4 "C" 4
#> 5 "D" 10
#> 6 "E" 3
#> 7 "F" 1
#> 8 "G" 10
#> 9 "H" 6
#> 10 "I" 1
#> # ... with 11 more rows
```

## 8.7 Case study - Canadian Prime Ministers

### 8.7.1 Introduction

In this case study we are interested in how long Canadian prime ministers lived, based on the year that they were born. We will scrape data from Wikipedia, clean it, and then make a graph.

The key library that we will use for scraping is `rvest`. This adds a lot of functions that will make life easier. That said, every time you scrape a website things will change. Each scrape will largely be bespoke, even if you can borrow some code from earlier projects that you have completed. It is completely normal to feel frustrated at times. It helps to begin with an end in mind.

To that end, let's generate some simulated data. Ideally, we want a table that has a row for each prime minister, a column for their name, and a column each for the birth and death years. If they are still alive, then that death year can be empty. We know that birth and death years should be somewhere between 1700 and 1990, and that death year should be larger than birth year. Finally, we also know that the years should be integers, and the names should be characters. So, we want something that looks roughly like this:

```
library(babynames)
library(tidyverse)

simulated_dataset <-
 tibble(prime_minister = sample(x = babynames %>% filter(prop > 0.01) %>%
 select(name) %>% unique() %>% unlist(),
```

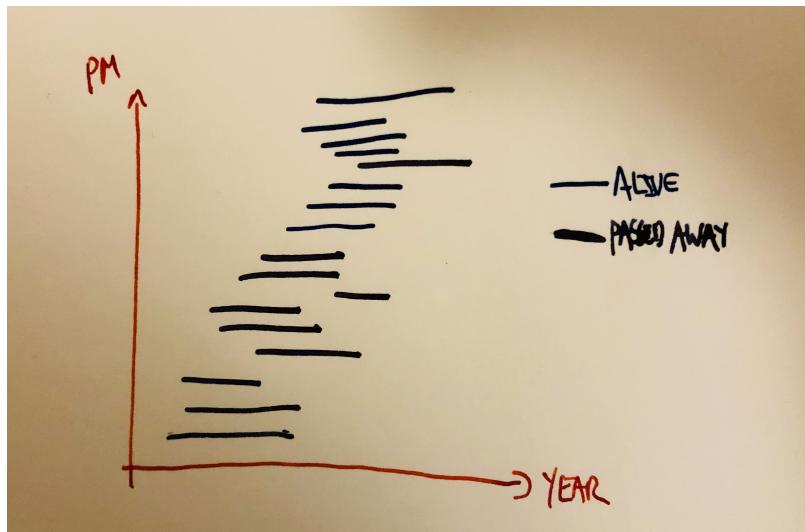
```

size = 10, replace = FALSE),
birth_year = sample(x = c(1700:1990), size = 10, replace = TRUE),
years_lived = sample(x = c(50:100), size = 10, replace = TRUE),
death_year = birth_year + years_lived) %>%
select(prime_minister, birth_year, death_year, years_lived) %>%
arrange(birth_year)

head(simulated_dataset)
#> # A tibble: 6 x 4
#> prime_minister birth_year death_year years_lived
#> <chr> <int> <int> <int>
#> 1 Tracy 1706 1800 94
#> 2 Ruth 1748 1828 80
#> 3 Michelle 1793 1857 64
#> 4 Andrew 1817 1879 62
#> 5 Donna 1846 1905 59
#> 6 Susan 1856 1948 92

```

One of the advantages of generating a simulated dataset is that if you are working in groups then one person can start making the graph, using the simulated dataset, while the other person gathers the data. In terms of a graph, we want something like Figure 8.7.



**FIGURE 8.7:** Sketch of planned graph.

### 8.7.2 Gather

We are starting with a question that is of interest, which how long each Canadian prime minister lived. As such, we need to identify a source of data. While there are likely to be plenty of data sources that have the births and deaths of each prime minister, we want one that we can trust, and as we are going to be scraping, we want one that has some structure to it. The Wikipedia page ([https://en.wikipedia.org/wiki/List\\_of\\_prime\\_ministers\\_of\\_Canada](https://en.wikipedia.org/wiki/List_of_prime_ministers_of_Canada)) fits both these criteria. As it is a popular page the information is more likely to be correct, and the data are available in a table.

We load the library and then we read in the data from the relevant page. The key function here is `read_html()`, which you can use in the same way as, say, `read_csv()`, except that it takes a html page as an input. Once you call `read_html()` then the page is downloaded to your own computer, and it is usually a good idea to save this, using `write_html()` as it is your raw data. Saving it also means that we don't have to keep visiting the website when we want to start again with our cleaning, and so it is part of being polite. However, it is likely not our property (in the case of Wikipedia, we might be okay), and so you should probably not share it.

```
library(rvest)
```

```
raw_data <- read_html("https://en.wikipedia.org/wiki/List_of_prime_ministers_of_Canada")
write_html(raw_data, "inputs/wiki/pms.html") # Note that we save the file as a html file.
```

### 8.7.3 Clean

Websites are made up of html, which is a markup language. We are looking for patterns in the mark-up that we can use to help us get closer to the data that we want. This is an iterative process and requires a lot of trial and error. Even simple examples will take time. You can look at the html by using a browser, right clicking, and then selecting `view page source`. Similarly, you could open the html file using a text editor.

#### 8.7.3.1 By inspection

We are looking for patterns that we can use to select the information that is of interest - names, birth year, and death year. When we look at the html it looks like there is something going on with `<tr>`, and then `<td>` (thanks to Thomas Rosenthal for identifying this). We select those nodes using `html_nodes()`, which takes the tags as an input. If you only want the first one then there is a singular version, `html_node()`.

```

Read in our saved data
raw_data <- read_html("inputs/wiki/pms.html")

We can parse tags in order
parse_data_inspection <-
 raw_data %>%
 html_nodes("tr") %>%
 html_nodes("td") %>%
 html_text() # html_text removes any remaining html tags

But this code does exactly the same thing - the nodes are just pushed into
the one function call
parse_data_inspection <-
 raw_data %>%
 html_nodes("tr td") %>%
 html_text()

head(parse_data_inspection)
#> [1] "Abbreviation key:"
#> [2] "No.: Incumbent number, Min.: Ministry, Refs: References\n"
#> [3] "Colour key:"
#> [4] "\n\n Liberal Party of Canada\n \n Historical Conservative parties (including Liberal-Con"
#> [5] "Provinces key:"
#> [6] "AB: Alberta, BC: British Columbia, MB: Manitoba, NS: Nova Scotia, ON: Ontario, QC: Quebec, "

```

At this point our data is in a character vector, we want to convert it to a table, and reduce the data down to just the information that we want. The key that is going to allow us to do this is the fact that there seems to be a blank line (which in html is denoted by \n) before the key information that we need. So, once we identify that line then we can filter to just the line below it!

```

parsed_data <-
 tibble(raw_text = parse_data_inspection) %>% # Convert the character vector to a table
 mutate(is_PM = if_else(raw_text == "\n\n", 1, 0), # Look for the blank line that is
 # above the row that we want
 is_PM = lag(is_PM, n = 1)) %>% # Identify the actual row that we want
 filter(is_PM == 1) # Just get the rows that we want

head(parsed_data)
#> # A tibble: 6 x 2
#> raw_text
#> <chr>
#> 1 "\nSir John A. MacDonald(1815–1891)MP for Kingston, ON\n"

```

`is_PM`  
`<dbl>`  
`1`

```
#> 2 "\nAlexander Mackenzie(1822-1892)MP for Lambton, ON\n"
#> 3 "\nSir John A. MacDonald(1815-1891)MP for Victoria, BC until 1882MP for~
#> 4 "\nSir John Abbott(1821-1893)Senator for Quebec\n"
#> 5 "\nSir John Thompson(1845-1894)MP for Antigonish, NS\n"
#> 6 "\nSir Mackenzie Bowell(1823-1917)Senator for Ontario\n"
```

### 8.7.3.2 Using the selector gadget

If you are comfortable with html then you might be able to see patterns, but one tool that may help is the SelectorGadget: <https://cran.r-project.org/web/packages/rvest/vignettes/selectorgadget.html>. This allows you to pick and choose the elements that you want, and then gives you the input to give to `html_nodes()` (Figure 8.8)

No.	Portrait	(Birth-Death) District	Term of office	Electoral mandates (Parliaments)	Political party	Min.	Refs
1		Sir John A. MacDonald (1815-1891) MP for Kingston, ON	1 July 1867 – 5 November 1873	<ul style="list-style-type: none"> <li>• Title created (no parliament)</li> <li>• 1867 election (1st Parliament)</li> <li>• 1872 election (2nd Parliament)</li> </ul>	Liberal-Conservative Party	1st	[2][5]
2		Alexander Mackenzie (1822-1892) MP for Lambton, ON	7 November 1873 – 8 October 1878	<ul style="list-style-type: none"> <li>• Appointment (2nd Parliament)[Min.]</li> <li>• 1874 election (3rd Parliament)</li> </ul>	Liberal Party Named leader in 1873	2nd	[6][7]
(1)		Sir John A. MacDonald (1815-1891) MP for Victoria, BC until 1882 MP for Carleton, ON until 1887 MP for Kingston, ON	17 October 1878 – 6 June 1891	<ul style="list-style-type: none"> <li>• 1878 election (4th Parliament)</li> <li>• 1882 election (5th Parliament)</li> <li>• 1887 election (6th Parliament)</li> <li>• 1891 election (7th Parliament)</li> </ul>	Liberal-Conservative Party	3rd	[8][9]
3		Sir John Abbott (1821-1892) Senator for Quebec	16 June 1891 – 24 November 1892	<ul style="list-style-type: none"> <li>• Appointment (7th Parliament)</li> </ul>	Liberal-Conservative Party	4th	[10] [11]
4		Sir John Thompson (1845-1894) MP for Antigonish, NS	5 December 1892 – 12 December 1894	<ul style="list-style-type: none"> <li>• Appointment (7th Parliament)</li> </ul>	Liberal-Conservative Party	5th	[12] [13]
5		Sir Mackenzie Bowell (1823-1917)	21 December 1894		Conservative Party		X

**FIGURE 8.8:** Using the Selector Gadget to identify the tag, as at 13 March 2020.

```
Read in our saved data
raw_data <- read_html("inputs/wiki/pms.html")

We can parse tags in order
parse_data_selector_gadget <-
 raw_data %>%
```

```

html_nodes("td:nth-child(3)") %>%
 html_text() # html_text removes any remaining html tags

head(parse_data_selector_gadget)
#> [1] "\nSir John A. MacDonald(1815-1891)MP for Kingston, ON\n"
#> [2] "\nAlexander Mackenzie(1822-1892)MP for Lambton, ON\n"
#> [3] "\nSir John A. MacDonald(1815-1891)MP for Victoria, BC until 1882MP for Carleton, ON until ."
#> [4] "\nSir John Abbott(1821-1893)Senator for Quebec\n"
#> [5] "\nSir John Thompson(1845-1894)MP for Antigonish, NS\n"
#> [6] "\nSir Mackenzie Bowell(1823-1917)Senator for Ontario\n"

```

In this case there is one prime minister - Robert Borden - who changed party and we would need to filter away that row: \nUnionist Party\n".

### 8.7.3.3 Clean data

Now that we have the parsed data, we need to clean it to match what we wanted. In particular we want a names column, as well as columns for birth year and death year. We will use `separate()` to take advantage of the fact that it looks like the dates are distinguished by brackets.

```

initial_clean <-
 parsed_data %>%
 separate(raw_text,
 into = c("Name", "not_name"),
 sep = "\\(",
 remove = FALSE) %>% # The remove = FALSE option here means that we
 # keep the original column that we are separating.
 separate(not_name,
 into = c("Date", "all_the_rest"),
 sep = "\\)",
 remove = FALSE)

head(initial_clean)
#> # A tibble: 6 x 6
#> raw_text Name not_name Date all_the_rest is_PM
#> <chr> <chr> <chr> <chr> <chr> <dbl>
#> 1 "\nSir John A. Ma~ "\nSir ~ "1815-1891)MP fo~ 1815~ "MP for Kingston, O~ 1
#> 2 "\nAlexander Mack~ "\nAlex~ "1822-1892)MP fo~ 1822~ "MP for Lambton, ON~ 1
#> 3 "\nSir John A. Ma~ "\nSir ~ "1815-1891)MP fo~ 1815~ "MP for Victoria, B~ 1
#> 4 "\nSir John Abbot~ "\nSir ~ "1821-1893)Senat~ 1821~ "Senator for Quebec~ 1
#> 5 "\nSir John Thomp~ "\nSir ~ "1845-1894)MP fo~ 1845~ "MP for Antigonish,~ 1
#> 6 "\nSir Mackenzie ~ "\nSir ~ "1823-1917)Senat~ 1823~ "Senator for Ontari~ 1

```

Finally, we need to clean up the columns.

```
cleaned_data <-
 initial_clean %>%
 select(Name, Date) %>%
 separate(Date, into = c("Birth", "Died"), sep = "-", remove = FALSE) %>% # The
PMs who have died have their birth and death years separated by a hyphen, but
you need to be careful with the hyphen as it seems to be a slightly odd type of
hyphen and you need to copy/paste it.
 mutate(Birth = str_remove(Birth, "b. ")) %>% # Alive PMs have slightly different format
 select(-Date) %>%
 mutate(Name = str_remove(Name, "\n")) %>% # Remove some html tags that remain
 mutate_at(vars(Birth, Died), ~as.integer(.)) %>% # Change birth and death to integers
 mutate(Age_at_Death = Died - Birth) %>% # Add column of the number of years they lived
 distinct() # Some of the PMs had two goes at it.

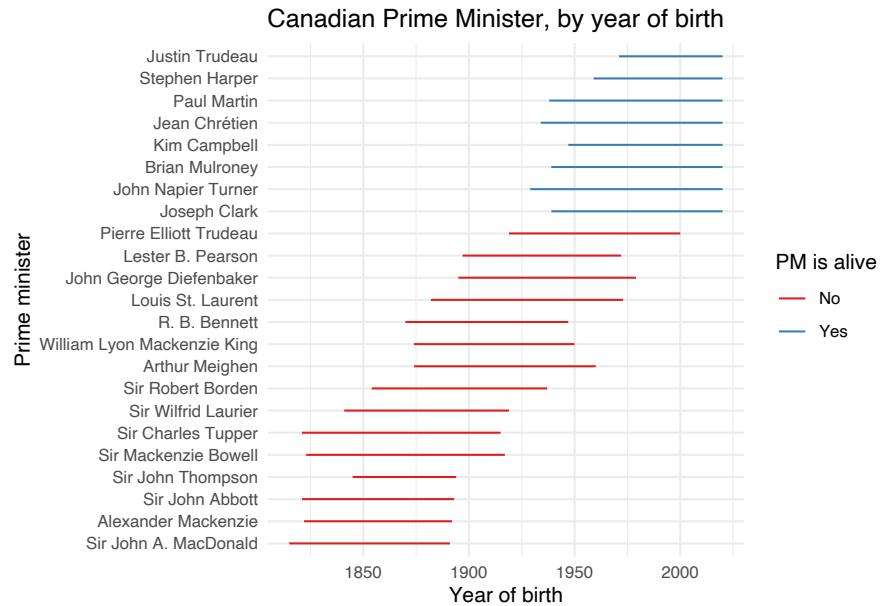
head(cleaned_data)
#> # A tibble: 6 x 4
#> Name Birth Died Age_at_Death
#> <chr> <int> <int> <int>
#> 1 Sir John A. MacDonald 1815 1891 76
#> 2 Alexander Mackenzie 1822 1892 70
#> 3 Sir John Abbott 1821 1893 72
#> 4 Sir John Thompson 1845 1894 49
#> 5 Sir Mackenzie Bowell 1823 1917 94
#> 6 Sir Charles Tupper 1821 1915 94
```

#### 8.7.4 Explore

At this point we'd like to make a graph that illustrates how long each prime minister lived. If they are still alive then we would like them to run to the end, but we would like to colour them differently.

```
cleaned_data %>%
 mutate(still_alive = if_else(is.na(Died), "Yes", "No"),
 Died = if_else(is.na(Died), as.integer(2020), Died)) %>%
 mutate(Name = as_factor(Name)) %>%
 ggplot(aes(x = Birth,
 xend = Died,
 y = Name,
 yend = Name,
 color = still_alive)) +
 geom_segment() +
```

```
labs(x = "Year of birth",
 y = "Prime minister",
 color = "PM is alive",
 title = "Canadian Prime Minister, by year of birth") +
theme_minimal() +
scale_color_brewer(palette = "Set1")
```



## 8.8 PDFs

### 8.8.1 Introduction

In contrast to an API, a PDF is usually only produced for human (not computer) consumption. The nice thing about PDFs is that they are static and constant. And it is nice that they make data available at all. But the trade-off is that:

- It is not overly useful to do larger-scale statistical analysis.
- We don't know how the PDF was put together so we don't know whether we can trust it.
- We can't manipulate the data to get results that we are interested in.

Indeed, sometimes governments publish data as PDFs because they don't ac-

tually want you to be able to analyse it! Being able to get data from PDFs opens up a large number of datasets for you, some of which we'll see in this chapter.

There are two important aspects to keep in mind when approaching a PDF with a mind to extracting data from it:

1. Begin with an end in mind. Planning and then literally sketching out what you want from a final dataset/graph/paper stops you wasting time and keeps you focused.
2. Start simple, then iterate. The quickest way to make a complicated model is often to first build a simple model and then complicate it. Start with just trying to get one page of the PDF working or even just one line. Then iterate from there.

In this chapter we start by walking through several examples and then go through three case studies of varying difficulty.

### 8.8.2 Getting started

Figure 8.9 is a PDF that consists of just the first sentence from Jane Eyre taken from Project Gutenberg Bronte (1847).

We will use the package `pdftools` Ooms (2019a) to get the text in this one page PDF into R.

```
install.packages("pdftools")
library(pdftools)
library(tidyverse)

first_example <- pdftools::pdf_text("inputs/pdfs/first_example.pdf")

first_example
#> [1] "There was no possibility of taking a walk that day.\n"

class(first_example)
#> [1] "character"
```

We can see that the PDF has been correctly read in, as a character vector.

We will now try a slightly more complicated example that consists of the first few paragraphs of Jane Eyre (Figure 8.10). Also notice that now we have the chapter heading as well.

We use the same function as before.

There was no possibility of taking a walk that day.

**FIGURE 8.9:** First sentence of Jane Eyre

## CHAPTER I

There was no possibility of taking a walk that day. We had been wandering, indeed, in the leafless shrubbery an hour in the morning; but since dinner (Mrs. Reed, when there was no company, dined early) the cold winter wind had brought with it clouds so sombre, and a rain so penetrating, that further out-door exercise was now out of the question.

I was glad of it: I never liked long walks, especially on chilly afternoons: dreadful to me was the coming home in the raw twilight, with nipped fingers and toes, and a heart saddened by the chidings of Bessie, the nurse, and humbled by the consciousness of my physical inferiority to Eliza, John, and Georgiana Reed.

The said Eliza, John, and Georgiana were now clustered round their mama in the drawing-room: she lay reclined on a sofa by the fireside, and with her darlings about her (for the time neither quarrelling nor crying) looked perfectly happy. Me, she had dispensed from joining the group; saying, "She regretted to be under the necessity of keeping me at a distance; but that until she heard from Bessie, and could discover by her own observation, that I was endeavouring in good earnest to acquire a more sociable and childlike disposition, a more attractive and sprightly manner—something lighter, franker, more natural, as it were—she really must exclude me from privileges intended only for contented, happy, little children."

"What does Bessie say I have done?" I asked.

"Jane, I don't like cavillers or questioners; besides, there is something truly forbidding in a child taking up her elders in that manner. Be seated somewhere; and until you can speak pleasantly, remain silent."

A breakfast-room adjoined the drawing-room, I slipped in there. It contained a bookcase: I soon possessed myself of a volume, taking care that it should be one stored with pictures. I mounted into the window-seat: gathering up my feet, I sat cross-legged, like a Turk; and, having drawn the red moreen curtain nearly close, I was shrined in double retirement.

Folds of scarlet drapery shut in my view to the right hand; to the left were the clear panes of glass, protecting, but not separating me from the drear November day. At intervals, while turning over the leaves of my book, I studied the aspect of that winter afternoon. Afar, it offered a pale blank of mist and cloud; near a scene of wet lawn and storm-beat shrub, with ceaseless rain sweeping away wildly before a long and lamentable blast.

**FIGURE 8.10:** First few paragraphs of Jane Eyre

```
second_example <- pdftools::pdf_text("inputs/pdfs/second_example.pdf")

second_example
#> [1] "CHAPTER I\nThere was no possibility of taking a walk that day. We had been wandering, inde

class(second_example)
#> [1] "character"
```

Again, we have a character vector. The end of each line is signalled by ‘\n’, but other than that it looks pretty good.

Finally, we consider the first two pages.

We use the same function as before.

```
third_example <- pdftools::pdf_text("inputs/pdfs/third_example.pdf")

third_example
#> [1] "CHAPTER I\nThere was no possibility of taking a walk that day. We had been wandering, inde
#> [2] "Of farthest Thule; and the Atlantic surge\nPours in among the stormy Hebrides."\n\nNor cou

class(third_example)
#> [1] "character"
```

Now, notice that the first page is the first element of the character vector and the second page is the second element.

As we’re most familiar with rectangular data we’ll try to get it into that format as quickly as possible. And then we can use our regular tools to deal with it.

First we want to convert the character vector into a tibble. At this point we may like to add page numbers as well.

```
jane_eyre <- tibble(raw_text = third_example,
 page_number = c(1:2))
```

We probably now want to separate the lines so that each line is an observation. We can do that by looking for the ‘\n’ remembering that we need to escape the backslash as it’s a special character.

```
jane_eyre <- separate_rows(jane_eyre, raw_text, sep = "\\\n", convert = FALSE)
head(jane_eyre)
#> # A tibble: 6 x 2
#> raw_text page_number
#> <chr> <dbl>
```

```
#> <chr>
#> 1 "CHAPTER I"
#> 2 "There was no possibility of taking a walk that day. We had been ~
#> 3 "leafless shrubbery an hour in the morning; but since dinner (Mrs~
#> 4 "company, dined early) the cold winter wind had brought with it c~
#> 5 "penetrating, that further out-door exercise was now out of the q~
#> 6 ""
```

	<int>
1	1
2	1
3	1
4	1
5	1
6	1

## 8.9 Case-study: US Total Fertility Rate, by state and year (2000-2018)

### 8.9.1 Introduction

If you're married to a demographer it is not too long until you are asked to look at a US Department of Health and Human Services Vital Statistics Report. In this case we are interested in trying to get the total fertility rate (the average number of births per woman assuming that woman experience the current age-specific fertility rates throughout their reproductive years)<sup>5</sup> for each state for nineteen years. Annoyingly, the US persists in only making this data available in PDFs, but it makes a nice case study.

In the case of the year 2000 the table that we are interested in is on page 40 of a PDF that is available [https://www.cdc.gov/nchs/data/nvsr/nvsr50/nvsr50\\_05.pdf](https://www.cdc.gov/nchs/data/nvsr/nvsr50/nvsr50_05.pdf) and it is the column labelled: "Total fertility rate" (Figure 8.11).

### 8.9.2 Begin with an end in mind

The first step when getting data out of a PDF is to sketch out what you eventually want. A PDF typically contains a lot of information, and so it is handy to be very clear about what you need. This helps keep you focused, and prevents scope creep, but it is also helpful when thinking about data checks. Literally write down on paper what you have in mind.

In this case, what is needed is a table with a column for state, year and TFR (Figure 8.12).

---

<sup>5</sup>And if you'd like to know more about this then I'd recommend starting a PhD with Monica Alexander<sup>6</sup>.

## 8.9 Case-study: US Total Fertility Rate, by state and year (2000-2018) 293

40 National Vital Statistics Report, Vol. 50, No. 5, Revised May 15, 2002

Table 10. Number of births, birth rates, fertility rates, total fertility rates, and birth rates for teenagers 15-19 years by age of mother:  
United States, each State and territory, 2000

[By place of residence. Birth rates are live births per 1,000 estimated population in each area; fertility rates are live births per 1,000 women aged 15-44 years estimated in each area; total fertility rates are sums of birth rates for 5-year age groups multiplied by 5; birth rates by age are live births per 1,000 women in specified age group estimated in each area]

State	Number of births	Birth rate	Fertility rate	Total fertility rate	Teenage birth rate		
					15-19 years		
					Total	15-17 years	18-19 years
United States <sup>1</sup>	4,058,814	14.7	67.5	2,130.0	48.5	27.4	79.2
Alabama	63,299	14.4	65.0	2,021.0	62.9	37.9	97.3
Alaska	9,974	16.0	74.6	2,437.0	42.4	23.6	69.4
Arizona	85,733	17.5	64.4	2,304.0	69.1	41.1	111.3
Arkansas	37,783	14.7	69.1	2,140.0	68.5	36.7	114.1
California	531,959	15.8	70.7	2,186.0	48.5	28.6	75.6
Colorado	65,438	15.8	73.5	2,300.0	49.2	28.6	79.8
Connecticut	49,096	13.0	61.2	1,931.5	31.9	16.9	55.3
Delaware	11,051	14.5	63.5	2,014.0	51.6	30.5	80.2
District of Columbia	7,666	14.8	63.0	1,975.5	80.7	60.7	101.8
Florida	204,125	13.3	66.9	2,157.5	52.6	29.7	88.0
Georgia	132,644	16.7	71.4	2,239.5	64.2	36.8	104.3
Hawaii	17,551	14.9	72.3	2,337.0	45.1	24.7	70.5
Iowa	20,566	16.0	74.3	2,310.0	45.1	21.3	72.8
Illinois	185,036	15.2	69.5	2,190.5	49.5	28.5	81.1
Indiana	87,699	14.7	66.8	2,109.0	50.3	26.2	85.9
Iowa	86,656	13.3	64.0	2,036.0	34.7	17.4	60.3
Kansas	39,666	14.9	69.2	2,205.0	45.3	22.4	78.5
Kentucky	56,026	14.1	63.6	1,992.5	55.3	29.2	92.2
Louisiana	67,898	15.5	69.1	2,128.5	62.1	36.3	97.1
Maine	13,603	10.8	49.5	1,611.5	28.7	13.4	52.8
Maryland	74,316	14.2	61.9	1,974.5	41.6	23.8	68.8
Massachusetts	81,614	13.2	59.2	1,799.0	27.1	15.0	44.9
Michigan	188,117	13.7	62.0	1,950.0	39.2	21.3	66.3
Minnesota	67,604	14.0	63.8	2,062.0	29.6	15.6	51.0
Mississippi	44,075	15.8	70.3	2,124.0	72.0	45.0	109.9
Missouri	78,675	13.9	64.0	2,036.0	48.8	26.5	62.2
Montana	10,957	12.3	61.3	2,003.0	35.8	19.1	60.8
Nebraska	24,646	14.8	68.9	2,209.0	37.2	19.3	62.7
Nevada	30,829	16.4	79.8	2,560.0	62.2	34.2	106.7
New Hampshire	14,609	12.0	52.2	1,864.0	23.4	9.8	45.4
New Jersey	115,632	14.1	65.8	2,086.0	31.7	17.0	54.9
New Mexico	27,223	15.6	72.7	2,313.0	66.2	40.2	105.1
New York	55,577	14.2	65.0	2,030.0	35.6	20.1	58.1
North Carolina	120,311	15.5	71.6	2,269.5	59.9	32.8	101.4
North Dakota	7,676	12.2	58.7	1,875.5	28.2	12.5	51.4
Ohio	153,527	13.8	63.6	1,953.0	45.6	24.1	77.2
Oklahoma	49,782	14.7	69.9	2,184.0	60.1	32.9	99.8
Oregon	45,804	13.7	65.8	2,086.0	43.2	23.5	72.8
Pennsylvania	146,281	12.2	58.2	1,868.0	35.2	19.6	58.8
Rhode Island	12,505	12.6	58.1	1,822.0	38.4	21.3	64.0
South Carolina	56,114	14.3	63.3	1,971.5	60.6	36.7	92.9
South Dakota	10,345	14.0	66.7	2,104.0	37.2	19.4	62.2
Tennessee	79,111	14.4	65.2	2,083.5	61.5	34.2	101.6
Texas	363,414	17.8	80.0	2,500.5	69.2	42.7	107.1
Utah	47,353	21.9	94.5	2,761.5	40.0	22.0	62.7
Vermont	6,109	10.9	48.3	1,580.0	24.1	10.6	44.5
Virginia	98,938	14.2	61.2	1,904.0	40.8	21.7	66.9
Washington	81,036	13.9	63.2	2,011.5	38.2	20.3	64.5
West Virginia	20,865	11.6	55.9	1,723.5	46.4	22.8	79.8
Wisconsin	69,326	13.1	60.4	1,840.5	34.5	16.3	58.8
Wyoming	6,253	13.0	62.7	1,976.5	40.8	19.0	73.4

FIGURE 8.11: Example Vital Statistics Report, from 2000

### 8.9.3 Start simple, then iterate.

There are 19 different PDFs, and we are interested in a particular column in a particular table in each of them. Unfortunately, there is nothing magical about what is coming. This first step requires working out the link for each, and the page and column name that is of interest. In the end, this looks like this.

```
monicas_data <- read_csv("inputs/tfr_tables_info.csv")
```

```
monicas_data %>%
 select(year, page, table, column_name, url) %>%
 gt()
```

year	page	table	column_name	url
------	------	-------	-------------	-----

2000	40	10	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr50/nvsr50_05.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr50/nvsr50_05.pdf</a>
2001	41	10	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr51/nvsr51_02.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr51/nvsr51_02.pdf</a>
2002	46	10	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr52/nvsr52_10.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr52/nvsr52_10.pdf</a>
2003	45	10	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr54/nvsr54_02.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr54/nvsr54_02.pdf</a>
2004	52	11	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr55/nvsr55_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr55/nvsr55_01.pdf</a>
2005	52	11	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr56/nvsr56_06.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr56/nvsr56_06.pdf</a>
2006	49	11	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr57/nvsr57_07.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr57/nvsr57_07.pdf</a>
2007	41	11	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr58/nvsr58_24.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr58/nvsr58_24.pdf</a>
2008	43	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr59/nvsr59_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr59/nvsr59_01.pdf</a>
2009	43	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr60/nvsr60_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr60/nvsr60_01.pdf</a>
2010	42	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr61/nvsr61_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr61/nvsr61_01.pdf</a>
2011	40	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr62/nvsr62_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr62/nvsr62_01.pdf</a>
2012	38	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr62/nvsr62_09.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr62/nvsr62_09.pdf</a>
2013	37	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr64/nvsr64_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr64/nvsr64_01.pdf</a>
2014	38	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr64/nvsr64_12.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr64/nvsr64_12.pdf</a>
2015	42	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr66/nvsr66_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr66/nvsr66_01.pdf</a>
2016	29	8	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_01.pdf</a>
2016	30	8	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_01.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_01.pdf</a>
2017	23	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_08-508.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_08-508.pdf</a>
2017	24	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_08-508.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr67/nvsr67_08-508.pdf</a>
2018	23	12	Total fertility rate	<a href="https://www.cdc.gov/nchs/data/nvsr/nvsr68/nvsr68_13-508.pdf">https://www.cdc.gov/nchs/data/nvsr/nvsr68/nvsr68_13-508.pdf</a>

The first step is to get some code that works for one of them. I'll step through the code in a lot more detail than normal because we're going to use these pieces a lot.

We will choose the year 2000. We first download the data and save it.

```
download.file(url = monicas_data$url[1],
 destfile = "inputs/pdfs/dhs/year_2000.pdf")
```

We now want to read the PDF in as a character vector.

```
dhs_2000 <- pdftools::pdf_text("inputs/pdfs/dhs/year_2000.pdf")
```

Convert it to a tibble, so that we can use familiar verbs on it.

```
dhs_2000 <- tibble(raw_data = dhs_2000)

head(dhs_2000)
#> # A tibble: 6 x 1
#> raw_data
#> <chr>
```

state	year	TFR
Alabama	2000	2.5
:	:	:
Arizona	2018	2.1
:	2000	
:	2018	

**FIGURE 8.12:** Desired output from the PDF

```
#> 1 "Volume 50, Number 5
#> 2 "2 National Vital Statistics Report, Vol. 50, No. 5, February 12, 2002\n\n~"
#> 3 "
#> 4 "4 National Vital Statistics Report, Vol. 50, No. 5, February 12, 2002\n\n~"
#> 5 "
#> 6 "6 National Vital Statistics Report, Vol. 50, No. 5, February 12, 2002\n\n ~"
```

Grab the page that is of interest (remembering that each page is an element of the character vector, hence a row in the tibble).

```
dhs_2000 <-
dhs_2000 %>%
slice(monicas_data$page[1])

head(dhs_2000)
#> # A tibble: 1 x 1
#> raw_data
#> <chr>
#> 1 "40 National Vital Statistics Report, Vol. 50, No. 5, Revised May 15, 20022\n~"
```

Now we want to separate the rows.

```
dhs_2000 <-
 dhs_2000 %>%
 separate_rows(raw_data, sep = "\n", convert = FALSE)

head(dhs_2000)
#> # A tibble: 6 x 1
#> raw_data
#> <chr>
#> 1 "40 National Vital Statistics Report, Vol. 50, No. 5, Revised May 15, 20022"
#> 2 ""
#> 3 "Table 10. Number of births, birth rates, fertility rates, total fertility ra~"
#> 4 "United States, each State and territory, 2000"
#> 5 "[By place of residence. Birth rates are live births per 1,000 estimated popu~"
#> 6 "estimated in each area; total fertility rates are sums of birth rates for 5~"
```

Now we are searching for patterns that we can use. (If you have a lot of tables that you are interested in grabbing from PDFs then it may also be worthwhile considering the `tabulizer` package which is specifically designed for that (Leeper, 2018). The issue is that it depends on Java and I always seem to run into trouble when I need to use Java so I avoid it when I can.)

Let's look at the first ten lines of content.

```
dhs_2000[13:22,]
#> # A tibble: 10 x 1
#> raw_data
#> <chr>
#> 1 "
#> 2 "
#> 3 "
#> 4 """
#> 5 """
#> 6 "United States 1 ~
#> 7 """
#> 8 "Alabama ~
#> 9 "Alaska ~
#> 10 "Arizona ~
```

It doesn't get much better than this:

1. We have dots separating the states from the data.
2. We have a space between each of the columns.

So we can now separate this in to separate columns. First we want to match

### 8.9 Case-study: US Total Fertility Rate, by state and year (2000-2018) 297

on when there is at least two dots (remembering that the dot is a special character and so needs to be escaped).

```
dhs_2000 <-
 dhs_2000 %>%
 separate(col = raw_data,
 into = c("state", "data"),
 sep = "\\.{2,}",
 remove = FALSE,
 fill = "right"
)

head(dhs_2000)
#> # A tibble: 6 x 3
#> raw_data state data
#> <chr> <chr> <chr>
#> 1 "40 National Vital Statistics Report~" "40 National Vital Statistics Rep~ <NA>
#> 2 "" "" <NA>
#> 3 "Table 10. Number of births, birth r~" "Table 10. Number of births, birt~ <NA>
#> 4 "United States, each State and terri~" "United States, each State and te~ <NA>
#> 5 "[By place of residence. Birth rates~" "[By place of residence. Birth ra~ <NA>
#> 6 "estimated in each area; total ferti~" "estimated in each area; total fe~ <NA>
```

We get the expected warnings about the top and the bottom as they don't have multiple dots.

(Another option here is to use the `pdf_data()` function which would allow us to use location rather than delimiters.)

We can now separate the data based on spaces. There is an inconsistent number of spaces, so we first squish any example of more than one space into just one.

```
dhs_2000 <-
 dhs_2000 %>%
 mutate(data = str_squish(data)) %>%
 tidyrr::separate(col = data,
 into = c("number_of_births",
 "birth_rate",
 "fertility_rate",
 "TFR",
 "teen_births_all",
 "teen_births_15_17",
 "teen_births_18_19"),
 sep = "\\s",
```

```

 remove = FALSE
)

head(dhs_2000)
#> # A tibble: 6 x 10
#> raw_data state data number_of_births birth_rate fertility_rate TFR
#> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 "40 National~" "40 Nati~" <NA> <NA> <NA> <NA> <NA>
#> 2 "" "" <NA> <NA> <NA> <NA> <NA>
#> 3 "Table 10. N~" "Table 1~" <NA> <NA> <NA> <NA> <NA>
#> 4 "United Stat~" "United ~" <NA> <NA> <NA> <NA> <NA>
#> 5 "[By place o~" "[By pla~" <NA> <NA> <NA> <NA> <NA>
#> 6 "estimated i~" "estimat~" <NA> <NA> <NA> <NA> <NA>
#> # ... with 3 more variables: teen_births_all <chr>, teen_births_15_17 <chr>,
#> # teen_births_18_19 <chr>

```

This is all looking fairly great. The only thing left is to clean up.

```

dhs_2000 <-
dhs_2000 %>%
 select(state, TFR) %>%
 slice(13:69) %>%
 mutate(year = 2000)

dhs_2000
#> # A tibble: 57 x 3
#> state TFR year
#> <chr> <chr> <dbl>
#> 1 " ~ <NA> 2000
#> 2 " ~ <NA> 2000
#> 3 " ~ <NA> 2000
#> 4 "" <NA> 2000
#> 5 "" <NA> 2000
#> 6 "United States 1 " 2,130~ 2000
#> 7 "" <NA> 2000
#> 8 "Alabama " 2,021~ 2000
#> 9 "Alaska " 2,437~ 2000
#> 10 "Arizona " 2,652~ 2000
#> # ... with 47 more rows

```

And we're done for that year. Now we want to take these pieces, put them into a function and then run that function over all 19 years.

#### 8.9.4 Iterating

##### 8.9.4.1 Get the PDFs

The first part is downloading each of the 19 PDFs that we need. We're going to build on the code that we used before. That code was:

```
download.file(url = monicas_data$url[1], destfile = "inputs/pdfs/dhs/year_2000.pdf")
```

To modify this we need:

1. To have it iterate through each of the lines in the dataset that contains our CSVs (i.e. where it says 1, we want 1, then 2, then 3, etc.).
2. Where it has a filename, we need it to iterate through our desired filenames (i.e. year\_2000, then year\_2001, then year\_2002, etc).
3. We'd like for it to do all of this in a way that is a little robust to errors. For instance, if one of the URLs is wrong or the internet drops out then we'd like it to just move onto the next PDF, and then warn us at the end that it missed one, not to stop. (This doesn't really matter because it's only 19 files, but it's pretty easy to find yourself doing this for thousands of files).

We will draw on the `purrr` package for this [Henry and Wickham \(2020\)](#).

```
library(purrr)
monicas_data <-
 monicas_data %>%
 mutate(pdf_name = paste0("inputs/pdfs/dhs/year_", year, ".pdf"))

purrr::walk2(monicas_data$url, monicas_data$pdf_name, purrr::safely(~download.file(.x, .y)))
```

What this code does it take the function `download.file()` and give it two arguments: `.x` and `.y`. The function `walk2()` then applies that function to the inputs that we give it, in this case the URLs column is the `.x` and the `pdf_name` column is the `.y`. Finally, the `safely()` function means that if there are any failures then it just moves onto the next file instead of throwing an error.

We now have each of the PDFs saved and we can move onto getting the data from them.

##### 8.9.4.2 Get data from the PDFs

Now we need to get the data from the PDFs. As before, we're going to build on the code that we used before. That code (overly condensed) was:

```
dhs_2000 <- pdftools::pdf_text("inputs/pdfs/dhs/year_2000.pdf")

dhs_2000 <-
 tibble(raw_data = dhs_2000) %>%
 slice(monicas_data$page[1]) %>%
 separate_rows(raw_data, sep = "\n", convert = FALSE) %>%
 separate(col = raw_data, into = c("state", "data"), sep = "\\.{2,}", remove = FALSE) %>%
 mutate(data = str_squish(data)) %>%
 separate(col = data,
 into = c("number_of_births", "birth_rate", "fertility_rate", "TFR", "teen_births_all",
 sep = "\s",
 remove = FALSE) %>%
 select(state, TFR) %>%
 slice(13:69) %>%
 mutate(year = 2000)

dhs_2000
```

There are a bunch of aspects here that have been hardcoded, but the first thing that we want to iterate is the argument to `pdf_text()`, then the number in in `slice()` will also need to change (that is doing the work to get only the page that we are interested in).

Two aspects are hardcoded, and these may need to be updated. In particular: 1) The separate only works if each of the tables has the same columns in the same order; and 2) the slice (which restricts the data to just the states) only works in this particular case. Finally, we add the year only at the end, whereas we'd need to bring that up earlier in the process.

We'll start by writing a function that will go through all the files, grab the data, get the page of interest, and then expand the rows. We'll then use a function from `purrr` to apply that function to all of the PDFs and to output a tibble.

```
get_pdf_convert_to_tibble <- function(pdf_name, page, year){

 dhs_table_of_interest <-
 tibble(raw_data = pdftools::pdf_text(pdf_name)) %>%
 slice(page) %>%
 separate_rows(raw_data, sep = "\n", convert = FALSE) %>%
 separate(col = raw_data,
 into = c("state", "data"),
 sep = "[\r|\n|\.]\\s+(:digit:)",
 remove = FALSE) %>%
```

```

 mutate(
 data = str_squish(data),
 year_of_data = year)

 print(paste("Done with", year))

 return(dhs_table_of_interest)
}

raw_dhs_data <- purrr::pmap_dfr(monicas_data %>% select(pdf_name, page, year),
 get_pdf_convert_to_tibble)

#> [1] "Done with 2000"
#> [1] "Done with 2001"
#> [1] "Done with 2002"
#> [1] "Done with 2003"
#> [1] "Done with 2004"
#> [1] "Done with 2005"
#> [1] "Done with 2006"
#> [1] "Done with 2007"
#> [1] "Done with 2008"
#> [1] "Done with 2009"
#> [1] "Done with 2010"
#> [1] "Done with 2011"
#> [1] "Done with 2012"
#> [1] "Done with 2013"
#> [1] "Done with 2014"
#> [1] "Done with 2015"
#> [1] "Done with 2016"
#> [1] "Done with 2016"
#> [1] "Done with 2017"
#> [1] "Done with 2017"
#> [1] "Done with 2018"

head(raw_dhs_data)
#> # A tibble: 6 x 4
#> raw_data state data year_of_data
#> <chr> <chr> <chr> <dbl>
#> 1 "40 National Vital Statistics~ "40 National Vital Statisti~ 50, ~ 2000
#> 2 "" "" <NA> 2000
#> 3 "Table 10. Number of births, ~ "Table 10. Number of births~ <NA> 2000
#> 4 "United States, each State an~ "United States, each State ~ <NA> 2000
#> 5 "[By place of residence. Birt~ "[By place of residence. Bi~ <NA> 2000
#> 6 "estimated in each area; tota~ "estimated in each area; to~ <NA> 2000

```

Now we need to clean up the state names and then filter on them.

```
states <- c("Alabama", "Alaska", "Arizona", "Arkansas", "California", "Colorado",
 "Connecticut", "Delaware", "Florida", "Georgia", "Hawaii", "Idaho",
 "Illinois", "Indiana", "Iowa", "Kansas", "Kentucky", "Louisiana",
 "Maine", "Maryland", "Massachusetts", "Michigan", "Minnesota",
 "Mississippi", "Missouri", "Montana", "Nebraska", "Nevada",
 "New Hampshire", "New Jersey", "New Mexico", "New York", "North Carolina",
 "North Dakota", "Ohio", "Oklahoma", "Oregon", "Pennsylvania",
 "Rhode Island", "South Carolina", "South Dakota", "Tennessee", "Texas",
 "Utah", "Vermont", "Virginia", "Washington", "West Virginia", "Wisconsin",
 "Wyoming", "District of Columbia")

raw_dhs_data <-
 raw_dhs_data %>%
 mutate(state = str_remove_all(state, "\\".),
 state = str_remove_all(state, "\u00a0"),
 state = str_remove_all(state, "\u0008"),
 state = str_replace_all(state, "United States 1", "United States"),
 state = str_replace_all(state, "United States1", "United States"),
 state = str_replace_all(state, "United States 2", "United States"),
 state = str_replace_all(state, "United States2", "United States"),
 state = str_replace_all(state, "United States\u00b2", "United States"),
) %>%
 mutate(state = str_squish(state)) %>%
 filter(state %in% states)

head(raw_dhs_data)
#> # A tibble: 6 x 4
#> raw_data state data year_of_data
#> <chr> <chr> <chr> <dbl>
#> 1 Alabama ~ Alabama 63,299 14.4 65.0 2~ 2000
#> 2 Alaska ~ Alaska 9,974 16.0 74.6 2,~ 2000
#> 3 Arizona ~ Arizona 85,273 17.5 84.4 2~ 2000
#> 4 Arkansas ~ Arkans~ 37,783 14.7 69.1 2~ 2000
#> 5 California ~ Califo~ 531,959 15.8 70.7 ~ 2000
#> 6 Colorado ~ Colora~ 65,438 15.8 73.1 2~ 2000
```

The next step is to separate the data and get the correct column from it. We're going to separate based on spaces once it is cleaned up.

```
raw_dhs_data <-
 raw_dhs_data %>%
 mutate(data = str_remove_all(data, "*")) %>%
```

```

separate(data, into = c("col_1", "col_2", "col_3", "col_4", "col_5",
 "col_6", "col_7", "col_8", "col_9", "col_10"),
 sep = " ",
 remove = FALSE)
head(raw_dhs_data)
#> # A tibble: 6 x 14
#> raw_data state data col_1 col_2 col_3 col_4 col_5 col_6 col_7 col_8 col_9
#> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 Alabama ..~ Alab~ 63,29~ 63,2~ 14.4 65.0 2,02~ 62.9 37.9 97.3 <NA> <NA>
#> 2 Alaska ...~ Alas~ 9,974~ 9,974 16.0 74.6 2,43~ 42.4 23.6 69.4 <NA> <NA>
#> 3 Arizona ..~ Ariz~ 85,27~ 85,2~ 17.5 84.4 2,65~ 69.1 41.1 111.3 <NA> <NA>
#> 4 Arkansas .~ Arka~ 37,78~ 37,7~ 14.7 69.1 2,14~ 68.5 36.7 114.1 <NA> <NA>
#> 5 California~ Cali~ 531,9~ 531,~ 15.8 70.7 2,18~ 48.5 28.6 75.6 <NA> <NA>
#> 6 Colorado .~ Colo~ 65,43~ 65,4~ 15.8 73.1 2,35~ 49.2 28.6 79.8 <NA> <NA>
#> # ... with 2 more variables: col_10 <chr>, year_of_data <dbl>

```

We can now grab the correct column.

```

tfr_data <-
raw_dhs_data %>%
 mutate(TFR = if_else(year_of_data < 2008, col_4, col_3)) %>%
 select(state, year_of_data, TFR) %>%
 rename(year = year_of_data)
head(tfr_data)
#> # A tibble: 6 x 3
#> state year TFR
#> <chr> <dbl> <chr>
#> 1 Alabama 2000 2,021.0
#> 2 Alaska 2000 2,437.0
#> 3 Arizona 2000 2,652.5
#> 4 Arkansas 2000 2,140.0
#> 5 California 2000 2,186.0
#> 6 Colorado 2000 2,356.5

```

Finally, we need to convert the case.

```

head(tfr_data)
#> # A tibble: 6 x 3
#> state year TFR
#> <chr> <dbl> <chr>
#> 1 Alabama 2000 2,021.0
#> 2 Alaska 2000 2,437.0
#> 3 Arizona 2000 2,652.5

```

```
#> 4 Arkansas 2000 2,140.0
#> 5 California 2000 2,186.0
#> 6 Colorado 2000 2,356.5

tfr_data <-
 tfr_data %>%
 mutate(TFR = str_remove_all(TFR, ","),
 TFR = as.numeric(TFR))

head(tfr_data)
#> # A tibble: 6 x 3
#> state year TFR
#> <chr> <dbl> <dbl>
#> 1 Alabama 2000 2021
#> 2 Alaska 2000 2437
#> 3 Arizona 2000 2652.
#> 4 Arkansas 2000 2140
#> 5 California 2000 2186
#> 6 Colorado 2000 2356.
```

And run some checks.

```
tfr_data %>%
skimr::skim()
```

In particular we want for there to be 51 states and for there to be 19 years.

And we're done.

```
head(tfr_data)
#> # A tibble: 6 x 3
#> state year TFR
#> <chr> <dbl> <dbl>
#> 1 Alabama 2000 2021
#> 2 Alaska 2000 2437
#> 3 Arizona 2000 2652.
#> 4 Arkansas 2000 2140
#> 5 California 2000 2186
#> 6 Colorado 2000 2356.

write_csv(tfr_data, "outputs/monicas_tfr.csv")
```

## 8.10 Semi-structured

### 8.10.1 JSON and XML

---

## 8.11 Optical Character Recognition

All of the above is predicated on having a PDF that is already ‘digitized’. But what if it is images? In that case you need to first use Optical Character Recognition (OCR). The go-to package is Tesseract ([Ooms, 2019c](#)). This is a R wrapper around the Tesseract open-source OCR engine.

Let’s see an example with a scan from the first page of Jane Eyre (Figure 8.13).

```
install.packages('tesseract')
library(tesseract)
text <- tesseract::ocr(here::here("figures/jane_scan.png"), engine = tesseract("eng"))
cat(text)
#> 1 THERE was no possibility of taking a walk that day. We had
#> been wandering, indeed, in the leafless shrubbery an hour in
#> the morning; but since dinner (Mrs Reed, when there was no com-
#> pany, dined early) the cold winter wind had brought with it clouds
#> so sombre, and a rain so penetrating, that further out-door exercise
#>
#> was now out of the question.
#>
#> I was glad of it: I never liked long walks, especially on chilly
#> afternoons: dreadful to me was the coming home in the raw twi-
#> light, with nipped fingers and toes, and a heart saddened by the
#> chidings of Bessie, the nurse, and humbled by the consciousness of
#> my physical inferiority to Eliza, John, and Georgiana Reed.
#>
#> The said Eliza, John, and Georgiana were now clustered round
#> their mama in the drawing-room: she lay reclined on a sofa by the
#> fireside, and with her darlings about her (for the time neither quar-
#> reling nor crying) looked perfectly happy. Me, she had dispensed
#> from joining the group; saying, ‘She regretted to be under the
#> necessity of keeping me at a distance; but that until she heard from
#> Bessie, and could discover by her own observation that I was
#> endeavouring in good earnest to acquire a more sociable and
#> child-like disposition, a more attractive and sprightly manner-
#> something lighter, franker, more natural as it were—she really
```

**I**T THERE was no possibility of taking a walk that day. We had been wandering, indeed, in the leafless shrubbery an hour in the morning; but since dinner (Mrs Reed, when there was no company, dined early) the cold winter wind had brought with it clouds so sombre, and a rain so penetrating, that further out-door exercise was now out of the question.

I was glad of it: I never liked long walks, especially on chilly afternoons: dreadful to me was the coming home in the raw twilight, with nipped fingers and toes, and a heart saddened by the chidings of Bessie, the nurse, and humbled by the consciousness of my physical inferiority to Eliza, John, and Georgiana Reed.

The said Eliza, John, and Georgiana were now clustered round their mama in the drawing-room: she lay reclined on a sofa by the fireside, and with her darlings about her (for the time neither quarrelling nor crying) looked perfectly happy. Me, she had dispensed from joining the group; saying, ‘She regretted to be under the necessity of keeping me at a distance; but that until she heard from Bessie, and could discover by her own observation that I was endeavouring in good earnest to acquire a more sociable and child-like disposition, a more attractive and sprightly manner—something lighter, franker, more natural as it were—she really must exclude me from privileges intended only for contented, happy, little children.’

‘What does Bessie say I have done?’ I asked.

‘Jane, I don’t like cavillers or questioners: besides, there is something truly forbidding in a child taking up her elders in that manner. Be seated somewhere; and until you can speak pleasantly, remain silent.’



**FIGURE 8.13:** Scan of first page of Jane Eyre.

```
#> must exclude me from privileges intended only for contented,
#> happy, littie children.'
#>
#> 'What does Bessie say I have done?' I asked.
#>
#> 'Jane, I don't like cavillers or questioners: besides, there is
#> something truly forbidding in a child taking up her elders in that
#> manner. Be seated somewhere; and until you can speak pleasantly,
#> remain silent.'
#>
#> . Bs aT sae] eae
#>
#> i; AN TCM TAN | Beal | Sees
#> a) } ; | i)
#> i i 4 | | A ae | i | eee eek?
#>
#> a an eames yi | bee
#> 1 nea elem | | oe pee
#> i i ae BC i i Hale
#> oul | ec hi
#> pan || i re a al! |
#>
#> ase } Oty 2 RIES ORT Sata ariel
#> SEEN BE - ==_
#> 15
```

---

## 8.12 Text

*Aspects of this section have been previously published.*

### 8.12.1 Introduction

Text data is all around us, and in many cases is some of the earliest types of data that we are exposed to. Recent increases in computational power, the development of new methods, and the enormous availability of text, means that there has been a great deal of interest in using text as data. Initial methods tend to focus, essentially, on converting text into numbers and then analysing them using traditional methods. More recent methods have begun to take advantage of the structure that is inherent in text, to draw additional meaning. The difference is perhaps akin to a child who can group similar colors, compared with a child who knows what objects are; although both crocodiles

and trees are green, and you can do something with that knowledge, you can do more by knowing that a crocodile could eat you, and a tree probably won't.

In this section we cover a variety of techniques designed to equip you with the basics of using text as data. One of the great things about text data is that it is typically not generated for the purposes of our analysis. That's great because it removes one of the unobservable variables that we typically have to worry about. The trade-off is that we typically have to do a bunch more work to get it into a form that we can work with.

### 8.12.2 Getting text data

Text as data is an exciting tool to apply. But many guides assume that you already have a nice dataset. Because we've focused on workflow in these notes, we know that's not likely to be true! In this section we will scrape some text from a website. We've already seen examples of scraping, but in general those were focused on exploiting tables in the website. Here we're going to instead focus on paragraphs of text, hence we'll focus on different html/css tags.

We're going to us the `rvest` package to make it easier to scrape data. We're also going to use the `purrr` package to apply a function to a bunch of different URLs. For those of you with a little bit of programming, this is an alternative to using a for loop. For those of you with a bit of CS, this is a package that adds functional programming to R.

```
library(rvest)
library(tidyverse)

Some websites
address_to_visit <- c("https://www.rba.gov.au/monetary-policy/rba-board-minutes/2020/2020-03-03.html"
 "https://www.rba.gov.au/monetary-policy/rba-board-minutes/2020/2020-02-04.html"
 "https://www.rba.gov.au/monetary-policy/rba-board-minutes/2019/2019-12-03.html"
 "https://www.rba.gov.au/monetary-policy/rba-board-minutes/2019/2019-11-05.html"
 "https://www.rba.gov.au/monetary-policy/rba-board-minutes/2019/2019-10-01.html"
 "https://www.rba.gov.au/monetary-policy/rba-board-minutes/2019/2019-09-03.html"
)

Save names
save_name <- address_to_visit %>%
 str_remove("https://www.rba.gov.au/monetary-policy/rba-board-minutes/") %>%
 str_remove(".html") %>%
 str_remove("20[:digit:]{{2}}/") %>%
 str_c("inputs/rba/", ., ".csv")
```

Create the function that will visit `address_to_visit` and save to `save_name` files.

```

visit_address_and_save_content <-
 function(name_of_address_to_visit,
 name_of_file_to_save_as) {
 # The function takes two inputs
 name_of_address_to_visit <- address_to_visit[1]
 name_of_file_to_save_as <- save_name[1]

 read_html(name_of_address_to_visit) %>% # Go to the website and read the html
 html_node("#content") %>% # Find the content part
 html_text() %>% # Extract the text of the content part
 write_lines(name_of_file_to_save_as) # Save as a text file
 print(paste("Done with", name_of_address_to_visit, "at", Sys.time()))
 # Helpful so that you know progress when running it on all the records
 Sys.sleep(sample(30:60, 1)) # Space out each request by somewhere between
 # 30 and 60 seconds each so that we don't overwhelm their server
 }

 # If there is an error then ignore it and move to the next one
visit_address_and_save_content <-
 safely(visit_address_and_save_content)

```

We now apply that function to our list of URLs.

```

Walk through the addresses and apply the function to each
walk2(address_to_visit,
 save_name,
 ~ visit_address_and_save_content(.x, .y))

```

The result is a bunch of files with saved text data.

In this case we used scraping, but there are, of course, many ways. We may be able to use APIs, for instance, In the case of the Airbnb dataset that we examined earlier in the notes. If you are lucky then it may simply be that there is a column that contains text data in your dataset.

### 8.12.3 Preparing text datasets

*This section draws on Sharla Gelfand's blog post, linked in the required readings.*

As much as I would like to stick with Australian economics and politics examples, I realise that this is probably only of limited interest to most of you. As such, in this section we will consider a dataset of Sephora reviews. Please read Sharla's blog post (<https://sharla.party/post/crying-sephora/>) for another take on this dataset.

In this section we assume that there is some text data that you have gathered. At this point we need to change it into a form that we can work with. For some applications this will be counts of words. For others it may be some variant of this. The dataset that we are going to use is from Sephora, was scraped by Connie<sup>7</sup> and I originally became aware of it because of Sharla<sup>8</sup>.

First let's read in the data.

```
This code is taken from https://sharla.party/post/crying-sephora/
library(dplyr)
library(jsonlite)
library(tidytext)

crying <-
 jsonlite::fromJSON("https://raw.githubusercontent.com/everestpipkin/datagardens/master/students/
 simplifyDataFrame = TRUE
)

crying <- as_tibble(crying[["reviews"]])

head(crying)
#> # A tibble: 6 x 6
#> date product_info$name $type $url review_body review_title stars
#> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 29 Mar 2016 Too Faced Bett~ Masc~ http~ "Now I can~ AWESOME 5 st~
#> 2 29 Sep 2016 Too Faced Bett~ Masc~ http~ "This hold~ if you're s~ 5 st~
#> 3 23 May 2017 Too Faced Bett~ Masc~ http~ "I just bo~ Hate it 1 st~
#> 4 15 Aug 2017 Too Faced Bett~ Masc~ http~ "To start ~ Nearly perf~ 5 st~
#> 5 21 Sep 2016 Too Faced Bett~ Masc~ http~ "This masc~ Amazing!! 5 st~
#> 6 30 May 2016 Too Faced Bett~ Masc~ http~ "Let's tal~ Tricky but ~ 5 st~
#> # ... with 1 more variable: userid <dbl>

names(crying)
#> [1] "date" "product_info" "review_body" "review_title" "stars"
#> [6] "userid"
```

We'll focus on the `review_body` variable and the number of stars `stars` that the reviewer gave. Most of them are 5 stars, so we'll just focus on whether or not the review is five stars.

<sup>7</sup>[https://twitter.com/crabbage\\_/](https://twitter.com/crabbage_/)

<sup>8</sup><https://sharla.party/post/crying-sephora/>

```

crying <-
 crying %>%
 select(review_body, stars) %>%
 mutate(stars = str_remove(stars, " stars?"), # The question mark at the end means it'll get rid of
 stars = as.integer(stars)
) %>%
 mutate(five_stars = if_else(stars == 5, 1, 0))

table(crying$stars)
#>
#> 1 2 3 4 5
#> 6 2 4 14 79

```

In this example we are going to split everything into separate words. When we do this it is just searching for a space, and so what other types of elements are going to be considered ‘words’?

```

crying_by_words <-
 crying %>%
 tidytext::unnest_tokens(word, review_body, token = "words")

head(crying_by_words)
#> # A tibble: 6 x 3
#> stars five_stars word
#> <int> <dbl> <chr>
#> 1 5 1 now
#> 2 5 1 i
#> 3 5 1 can
#> 4 5 1 cry
#> 5 5 1 all
#> 6 5 1 i

```

We now want to count the number of times each word is used by each of the star classifications.

```

crying_by_words <-
 crying_by_words %>%
 count(stars, word, sort = TRUE)

head(crying_by_words)
#> # A tibble: 6 x 3
#> stars word n
#> <int> <chr> <int>
#> 1 1 now 1
#> 2 1 i 1
#> 3 1 can 14
#> 4 1 cry 6
#> 5 1 all 2
#> 6 1 i 6

```

```
#> 1 5 i 348
#> 2 5 and 249
#> 3 5 the 239
#> 4 5 it 211
#> 5 5 a 193
#> 6 5 this 178

crying_by_words %>%
 filter(stars == 1) %>%
 head()

#> # A tibble: 6 x 3
#> stars word n
#> <int> <chr> <int>
#> 1 1 the 39
#> 2 1 i 24
#> 3 1 and 21
#> 4 1 it 21
#> 5 1 to 19
#> 6 1 my 16
```

So you can see that the most popular word for five-star reviews is ‘i’, and that the most popular word for one star reviews is ‘the’.

At this point, we can use the data to do a whole bunch of different things, but one nice measure to look at is term frequency e.g. in this case how many times a word used in reviews with a particular star rating. The issue is that there are a lot of words that are commonly used regardless of context. As such, we may also like to look at the inverse document frequency in which we ‘penalise’ words that occur in many particular star ratings. For instance, ‘the’ probably occurs in both one star and five star reviews and so its idf is lower than ‘hate’ which probably only occurs in one star reviews. The term frequency-inverse document frequency (tf-idf) is then the product of these.

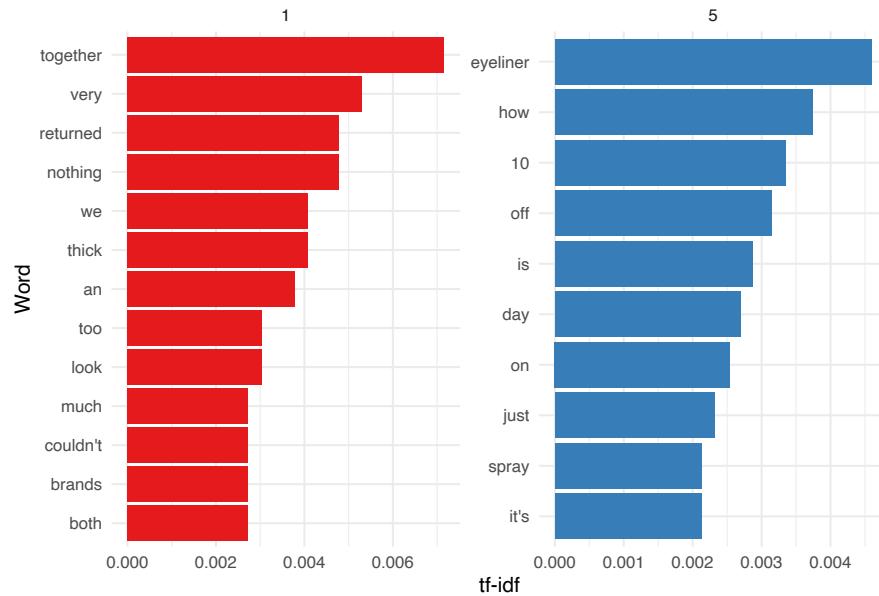
We can create this value using the `bind_tf_idf()` function from the `tidytext` package, and this will create a bunch of new columns, one for each word and star combination.

```
This code, and the one in the next block, is from Julia Silge: https://juliasilge.com/blog/sherlock/
crying_by_words_tf_idf <-
 crying_by_words %>%
 bind_tf_idf(word, stars, n) %>%
 arrange(-tf_idf)

head(crying_by_words_tf_idf)
```

```
#> # A tibble: 6 x 6
#> stars word n tf idf tf_idf
#> <int> <chr> <int> <dbl> <dbl> <dbl>
#> 1 2 below 1 0.00826 1.61 0.0133
#> 2 2 boy 1 0.00826 1.61 0.0133
#> 3 2 choice 1 0.00826 1.61 0.0133
#> 4 2 contrary 1 0.00826 1.61 0.0133
#> 5 2 exceptionally 1 0.00826 1.61 0.0133
#> 6 2 migrates 1 0.00826 1.61 0.0133
```

```
crying_by_words_tf_idf %>%
 group_by(stars) %>%
 top_n(10) %>%
 ungroup %>%
 mutate(word = reorder_within(word, tf_idf, stars)) %>%
 mutate(stars = as_factor(stars)) %>%
 filter(stars %in% c(1, 5)) %>%
 ggplot(aes(word, tf_idf, fill = stars)) +
 geom_col(show.legend = FALSE) +
 facet_wrap(vars(stars), scales = "free") +
 scale_x_reordered() +
 coord_flip() +
 labs(x = "Word",
 y = "tf-idf") +
 theme_minimal() +
 scale_fill_brewer(palette = "Set1")
```



## 8.13 Exercises and tutorial

### 8.13.1 Exercises

1. In your own words, what is an API (write a paragraph or two)?
2. Find two APIs and discuss how you could use them to tell interesting stories (write a paragraph or two for each).
3. Find two APIs that have an R packages written around them. How could you use these to tell interesting stories? (Write a paragraph or two for each.)
4. What is the main argument to `httr::GET()` (pick one)?
  - a. ‘url’
  - b. ‘website’
  - c. ‘domain’
  - d. ‘location’
5. Name three reasons why we should be respectful when getting scraping data from websites (write a paragraph or two).
6. What features of a website do we typically take advantage of when we parse the code (select all)?
  - a. HTML/CSS mark-up.
  - b. Cookies.
  - c. Facebook beacons.

- d. Code comments.
7. What are three advantages and three disadvantages of scraping compared with using an API (write a paragraph or two)?
  8. What are three delimiters that could be useful when trying to bring order to the PDF that you read in as a character vector (write a paragraph or two)?
  9. What do I need to put inside “SOMETHING\\_HERE” if I want to match regular expressions for a full stop i.e. “.” (hint: see the ‘strings’ cheat sheet) (pick one)?
    - a. .
    - b. \.
    - c. \\.
    - d. \\\.
  10. Name three reasons for sketching out what you want before starting to try to extract data from a PDF (write a paragraph or two for each).
  11. If you are interested in demographic data then what are three checks that you might like to do? What are three if you are interested in economic data such as GDP, interest rates, and exchange rates? (Write an explanation for each.)
  12. What does the `purrr` package do (select all)?
    - a. Enhances R’s functional programming toolkit.
    - b. Makes loops easier to code and read.
    - c. Checks the consistency of datasets.
    - d. Identifies issues in data structures and proposes replacements.
  13. Which of these are functions from the `purrr` package (select all)?
    - a. `map()`
    - b. `walk()`
    - c. `run()`
    - d. `safely()`
  14. Why should we use `safely()` when scraping data (pick one)?
    - a. To protect us from hackers.
    - b. To avoid side effects of pages with issues.
    - c. To slow down our scraping to an appropriate speed.
  15. What are some principles to follow when scraping (select all)?
    - a. Avoid it if possible
    - b. Follow the site’s guidance
    - c. Slow down
    - d. Use a scalpel not an axe.
  16. What is a robots.txt file (pick one)?
    - a. The instructions that Frankenstein followed.
    - b. Notes that web scrapers should follow when scraping.
  17. What is the html tag for an item in list (pick one)?
    - a. `li`

- b. `body`
  - c. `b`
  - d. `em`
18. If I have the following text data ‘rohan\_alexander’ in a column called ‘names’ and want to split it into first name and surname based on the underbar what function should I use (pick one)?
- a. `separate()`
  - b. `slice()`
  - c. `spacing()`
  - d. `text_to_columns()`

### 8.13.2 Tutorial

Gather some data yourself using a method that is introduced here - APIs directly or via a wrapper package, web scraping, PDF parsing, OCR, or text. Write a few paragraphs about the data source, what you gathered, and how you went about it. What took longer than you expected? When did it become fun? What would you do differently next time you do this? Please include a link to your GitHub repo so I can see the code, but it won’t be strictly marked - this is more about encouraging you to have a go. (Start with something tiny and very specific, get that working, and then increase the scope - almost everything will be more difficult and time-consuming than you think - and don’t forget to plan it out before you start.)

# 9

---

## Hunt data

---

**STATUS:** Under construction.

### Required reading

- Banerjee, Abhijit Vinayak, 2020, ‘Field Experiments and the Practice of Economics’, *American Economic Review*, Vol. 110, No. 7, pp. 1937-1951.
- Berry, Donald, 1989, ‘Comment: Ethics and ECMO’, *Statistical Science*, Vol 4, No 4, pp. 306-310.
- Duflo, Esther, 2020, ‘Field Experiments and the Practice of Policy’, *American Economic Review*, Vol. 110, No. 7, pp. 1952-1973 (or watch the speech detailed below).
- Fisher, Ronald, 1935, *The Design of Experiments*, pp. 20-29, <https://archive.org/details/in.ernet.dli.2015.502684/page/n33/mode/2up>.
- Fry, Hanna, 2020, ‘Experiments on Trial’, *The New Yorker*, 2 March, pp. 61-65, <https://www.newyorker.com/magazine/2020/03/02/big-tech-is-testing-you>.
- Gertler, Paul, Sebastian Martinez, Patrick Premand, Laura Rawlings, and Christel Vermeersch, *Impact Evaluation in Practice*, Chapters 3 and 4, <https://www.worldbank.org/en/programs/sief-trust-fund/publication/impact-evaluation-in-practice>.
- Hill, Austin Bradford, 1965, ‘The Environment and Disease: Association or Causation?’, *Proceedings of the Royal Society of Medicine*, 58, 5, 295-300.
- Kohavi, Ron and Stefan Thomke, 2017, ‘The Surprising Power of Online Experiments’, *Harvard Business Review*, September-October, <https://hbr.org/2017/09/the-surprising-power-of-online-experiments>.
- Kohavi, Ron, Diane Tang, and Ya Xu, 2020, *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing*, Cambridge University Press. (This sounds like a lot, but it’s a light book - it’s more about providing examples of issues to think about.) (Freely available through the U of T library.)
- Taback, Nathan, 2020, *Design of Experiments and Observational Studies*, Chapter 8 - Completely Randomized Designs: Comparing More Than Two Treatments, <https://scidesign.github.io/designbook/completely-randomized-designs-comparing-more-than-two-treatments.html>.
- Taylor, Sean, Dean Eckles, 2017, ‘Randomized experiments to detect and

- estimate social influence in networks', *arXiv*, <https://arxiv.org/abs/1709.09636v1>.
- Ware, James H., 1989, 'Investigating Therapies of Potentially Great Benefit: ECMO', *Statistical Science*, Vol 4, No 4, pp. 298-306.
  - Wu, Changbao and Mary E. Thompson, 2020, *Sampling Theory and Practice*, Springer, Chapters 1-3, and 5 (freely available through the U of T library).

### Required viewing

- Ge, Kathy, 2021, 'Experimentation and product design at Uber', *Toronto Data Workshop*, 4 February, <https://youtu.be/UYzXELJTovg>.
- Register, Yim, 2020, 'Introduction to Sampling and Randomization', *Online Causal Inference Seminar*, 14 November, <https://youtu.be/U272FFxG8LE>.
- Xu, Ya, 2020, 'Causal inference challenges in industry, a perspective from experiences at LinkedIn', *Online Causal Inference Seminar*, 16 July, <https://youtu.be/OoKsLAvyIYA>.

### Recommended reading

- Angrist, Joshua D., and Jörn-Steffen Pischke, 2008, *Mostly harmless econometrics: An empiricist's companion*, Princeton University Press, Chapter 2.
- Banerjee, Abhijit Vinayak, Esther Duflo, Rachel Glennerster, and Dhruba Kothari, 2010, 'Improving immunisation coverage in rural India: clustered randomised controlled evaluation of immunisation campaigns with and without incentives', *BMJ*, 340, c2220.
- Beaumont, Jean-François, 2020, 'Are probability surveys bound to disappear for the production of official statistics?', *Survey Methodology*, 46 (1), Statistics Canada, Catalogue No. 12-001-X.
- Christian, Brian, 2012, 'The A/B Test: Inside the Technology That's Changing the Rules of Business', *Wired*, 25 April, <https://www.wired.com/2012/04/ff-abtesting/>.
- Dablander, Fabian, 2020, "An Introduction to Causal Inference", *PsyArXiv*, 13 February, doi:10.31234/osf.io/b3fkw, <https://psyarxiv.com/b3fkw>.
- Deaton, Angus, 2010, 'Instruments, Randomization, and Learning about Development', *Journal of Economic Literature*, vol. 48, no. 2, pp. 424-455.
- Duflo, Esther, Rachel Glennerster, and Michael Kremer, 2007, 'Using Randomization In Development Economics Research: A Toolkit', <https://economics.mit.edu/files/806>.
- Gordon, Brett R., Florian Zettelmeyer, Neha Bhargava, and Dan Chapsky, 2019, 'A Comparison of Approaches to Advertising Measurement: Evidence from Big Field Experiments at Facebook', *Marketing Science*, Vol. 38, No. 2, March–April, pp. 193–225.
- Groves, Robert M., 2011, 'Three Eras of Survey Research', *Public Opinion Quarterly*, 75 (5), pp. 861–871, <https://doi.org/10.1093/poq/nfr057>.
- Hillygus, D. Sunshine, 2011, 'The evolution of election polling in the United States', *Public Opinion Quarterly*, 75 (5), pp. 962-981.

- Imai, Kosuke, 2017, *Quantitative Social Science: An Introduction*, Princeton University Press, Ch 2.3, 2.4, 4.3.
- Jeffries, Adrienne, Leon Yin, and Surya Mattu, 2020, ‘Swinging the Vote?’, *The Markup*, 26 February, <https://themarkup.org/google-the-giant/2020/02/26/wheres-my-email>.
- Kohavi, Ron, Alex Deng, Brian Frasca, Roger Longbotham, Toby Walker, and Ya Xu. 2012. Trustworthy online controlled experiments: five puzzling outcomes explained. In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD ’12). Association for Computing Machinery, New York, NY, USA, 786–794. DOI:<https://doi.org/10.1145/2339530.2339653>
- Landesberg, Eddie, Molly Davies, and Stephanie Yee, 2019, ‘Want to make good business decisions? Learn causality’, *MultiThreaded, Stitchfix blog*, 19 December, <https://multithreaded.stitchfix.com/blog/2019/12/19/good-marketing-decisions/>.
- Levay, Kevin E., Jeremy Freese, and James N. Druckman, 2016, ‘The demographic and political composition of Mechanical Turk samples’, *Sage Open*, 6 (1), 2158244016636433.
- Lewis, Randall A., and David H. Reiley, 2014 ‘Online ads and offline sales: Measuring the effects of retail advertising via a controlled experiment on Yahoo!’, *Quantitative Marketing and Economics*, Vol 12, pp. 235–266.
- Mullinix, Kevin J., Leeper, Thomas J., Druckman, James N. and Freese, Jeremy, 2015, ‘The generalizability of survey experiments’, *Journal of Experimental Political Science*, 2 (2), pp. 109-138.
- Novak, Greg, Sven Schmit, and Dave Spiegel, 2020, Experimentation with resource constraints, 18 November, StitchFix Blog, <https://multithreaded.stitchfix.com/blog/2020/11/18/virtual-warehouse/>.
- Prepared for the AAPOR Executive Council by a Task Force operating under the auspices of the AAPOR Standards Committee, with members including:, Reg Baker, Stephen J. Blumberg, J. Michael Brick, Mick P. Couper, Melanie Courtright, J. Michael Dennis, Don Dillman, Martin R. Frankel, Philip Garland, Robert M. Groves, Courtney Kennedy, Jon Krosnick, Paul J. Lavrakas, Sunghee Lee, Michael Link, Linda Piekarski, Kumar Rao, Randall K. Thomas, Dan Zahs, 2010, ‘Research Synthesis: AAPOR Report on Online Panels’, *Public Opinion Quarterly*, 74 (4), pp. 711–781, <https://doi.org/10.1093/poq/nfq048>.
- Ryan, A. C., A. R. MacKenzie, S. Watkins, and R. Timmis, 2012, ‘World War II contrails: a case study of aviation-induced cloudiness’, *International journal of climatology*, 32, no. 11, pp. 1745-1753.
- Said, Chris, 2020, ‘Optimizing sample sizes in A/B testing, Part I: General summary’, 10 January, <https://chris-said.io/2020/01/10/optimizing-sample-sizes-in-ab-testing-part-I/>. (See also parts 2 and 3).
- Stolberg, Michael, 2006, ‘Inventing the randomized double-blind trial: the Nuremberg salt test of 1835’, *Journal of the Royal Society of Medicine*, 99, no. 12, pp. 642-643.

- Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel, 2019, popular science background, <https://www.nobelprize.org/uploads/2019/10/popular-economicsciencesprize2019-2.pdf>.
- Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel, 2019, scientific background, <https://www.nobelprize.org/uploads/2019/10/advanced-economicsciencesprize2019.pdf>.
- Taddy, Matt, 2019, *Business Data Science*, Chapter 5.
- Urban, Steve, Rangarajan Sreenivasan, and Vineet Kannan, 2016, ‘It’s All A/Bout Testing: The Netflix Experimentation Platform’, *Netflix Technology Blog*, 29 April, <https://netflixtechblog.com/its-all-a-bout-testing-the-netflix-experimentation-platform-4e1ca458c15>.
- VWO, ‘A/B Testing Guide’, <https://vwo.com/ab-testing/>.
- Yeager, David S., Jon A. Krosnick, LinChiat Chang, Harold S. Javitz, Matthew S. Levendusky, Alberto Simpser, Rui Wang, 2011, ‘Comparing the Accuracy of RDD Telephone Surveys and Internet Surveys Conducted with Probability and Non-Probability Samples’, *Public Opinion Quarterly*, 75 (4), pp. 709–747, <https://doi.org/10.1093/poq/nfr020>.
- Yin, Xuan and Ercan Yildiz, 2020, ‘The Causal Analysis of Cannibalization in Online Products’, *Code as Craft, Etsy blog*, 24 February, <https://codeascraft.com/2020/02/24/the-causal-analysis-of-cannibalization-in-online-products/>.

### Recommended listening

- Galef, Julia, 2020, ‘Episode 246: Deaths of despair / Effective altruism (Angus Deaton)’, *Rationally Speaking*, from 35:30 through to the end, available at: <http://rationallyspeakingpodcast.org/show/episode-246-deaths-of-despair-effective-altruism-angus-deato.html>.

### Recommended viewing

- Duflo, Esther, 2020, ‘Interview with Esther Duflo’, 12 October, *Online Causal Inference Seminar*, <https://youtu.be/WWW9q3oMYxU>.
- Duflo, Esther, 2019, ‘Nobel Prize Lecture’, 8 December 2019, Stockholm: <https://www.nobelprize.org/prizes/economic-sciences/2019/duflo/lecture/>.
- Tipton, Elizabeth, 2020, ‘Will this Intervention Work in this Population? Designing Randomized Trials for Generalization’, *Online Causal Inference Seminar*, 14 April, <https://youtu.be/HYP32wzEZMA>.

### Key concepts/skills/etc

- Treatment and control groups.
- Internal and external validity.
- Average treatment effect.
- Generating simulated datasets.
- Defining populations, frames and samples.
- Distinguishing probability and non-probability sampling

- Distinguishing strata and clusters.

**Key libraries**

- broom
- ggplot2
- tidyverse

**Key functions/etc**

- aov()
  - rnorm()
  - sample()
  - t.test()
- 

## 9.1 Experiments and randomised controlled trials

### 9.1.1 Introduction

*First a note on Ronald Fisher and Francis Galton. Fisher and Galton are the intellectual grandfathers of much of the work that we cover. In some cases it is directly their work, in other cases it is work that built on their contributions. Both of these men believed in eugenics, amongst other things that are generally reprehensible.*

This chapter is about experiments. This is a situation in which we can explicitly control and vary some aspects. The advantage of this is that identification should be clear. There is a treatment group that is treated and a control group that is not. These are randomly split. And so if they end up different then it must be because of the treatment. Unfortunately, life is rarely so smooth. Arguing about how similar the treatment and control groups were tends to carry on indefinitely, because our ability to speak to internal validity affects our ability to speak to external validity.

It's also important to note that the statistics of this were designed in agricultural settings 'does fertilizer work?', etc. In those settings you can more easily divide a field into 'treated' and 'non-treated', and the magnitude of the effect is large. In general, these same statistical approaches are still used today (especially in the social sciences) but often inappropriately. If you hear someone talking about 'having enough power' and similar phrases, then it's *not* necessarily that they're *not* right, but it usually pays to take a step back and really think about what is being done and whether they really know what they're doing.

### 9.1.2 Motivation and notation

---

Never forget: if your sampling is in any way non-representative, your observe[d] data is not sufficient for population estimates. You *must* deal with design, sampling issues, data quality, and misclassification. Otherwise you'll just be wrong.

Dan Simpson, 30 January 2020<sup>1</sup>.

---

When Monica and I moved to San Francisco, the Giants immediately won the baseball, and the Warriors began a historic streak. We moved to Chicago and the Cubs won the baseball for the first time in a hundred years. We then moved to Massachusetts, and the Patriots won the Super Bowl again and again and again. Finally, we moved to Toronto, and the Raptors won the basketball. Should a city pay us to live there or could their funds be better spent elsewhere?

One way to get at the answer would be to run an experiment. Make a list of the North American cities with major sports teams, and then roll a dice and send us to live there for a year. If we had enough lifetimes, then we could work it out. The fundamental issue is that we cannot both live in a city and not live in a city. Experiments and randomised controlled trials are circumstances in which we try to randomly allocate some treatment, so as to have a belief that everything else was constant (or at least ignorable).

In the words of Hernan and Robins (2020, p. 3) an action,  $A$ , is also known ‘as an intervention, an exposure, or a treatment.’ I’ll typically use ‘treated/control’ language, reflecting whether an action was imposed or not. That treatment random variable will typically be binary, that is 0 or 1, ‘treated’ or ‘not treated/control/comparison’. We’ll then typically have some outcome random variable,  $Y$ , which will typically be binary, able to be made binary, or continuous, although we’ll touch on other options. An example of a binary outcome could be vote choice - ‘Conservative’ vs ‘Not Conservative’ - noticing there that I grouped all the other parties into simply ‘Not Conservative’ to force the binary outcome.

Further following Hernan and Robins (2020, p. 4), but in the notation of Gertler et al. (2016, p. 48) we describe a treatment as ‘causal’ when  $(Y|a = 0) \neq (Y|a = 1)$ . As discussed above, the fundamental problem of causal inference is that we cannot both treat and control the one individual.

---

<sup>1</sup>[https://twitter.com/dan\\_p\\_simpson/status/1222924987667046400](https://twitter.com/dan_p_simpson/status/1222924987667046400)

So when we want to know the effect of the treatment, we need to compare it with the counterfactual, which is what would have happened if the individual were not treated. So causal inference turns out to be fundamentally a missing data problem.<sup>2</sup>

To quote from Gertler et al. (2016, p.48), in the context of evaluating income in response to an intervention program:

---

To put it another way, we would like to measure income at the same point in time for the same unit of observation (a person, in this case), but in two different states of the world. If it were possible to do this, we would be observing how much income the same individual would have had at the same point in time both with and without the program, so that the only possible explanation for any difference in that person's income would be the program. By comparing the same individual with herself at the same moment, we would have managed to eliminate any outside factors that might also have explained the difference in outcomes. We could then be confident that the relationship between the vocational training program and the change in income is causal... [A] unit either participated in the program or did not participate. The unit cannot be observed simultaneously in two different states (in other words, with and without the program).

---

As we cannot compare treatment and control in one particular individual, we instead compare the average of two groups - all those treated and all those not. We are looking to estimate the counterfactual. We usually consider a default that there's no effect and we require evidence for us to change our mind. As we're interested in what is happening in groups, we turn to expectations, and notions of probability to express ourselves. Hence, we'll make claims that talk, on average. Maybe wearing fun socks really does make you have a lucky day, but on average across the population, it's probably not the case.<sup>3</sup>

It's worth pointing out that we don't just have to be interested in the average

---

<sup>2</sup>There's a joke in statistics, okay, well, TBH, I have a joke about statistics, and it's that at some point every professor is like '... and so X really just boils down to a missing data problem' and it's funny because, that's kind of the fundamental issue of statistics, we'd not really need the science if we had all the data. In hindsight, this is not really a joke, but I'm a father now so I'll just lean into it.

<sup>3</sup>As someone who oddly is somewhat superstitious, believes fully in the irony gods, and does have a pair of lucky, fun, socks, this example was not randomly chosen.

effect. We may consider the median, or variance, or whatever. Nonetheless, if we were interested in the average effect, then one way to proceed would be to divide the dataset into two - treated and not treated - have an effect column of 0s and 1s, sum the column and divide it by the length of the column, and then look at the ratio. This would be an estimator, which is a way of putting together a guess of something of interest. The estimand is the thing of interest, in this case the average effect, and the estimate is whatever our guess turns out to be. To give another example, following [Gelman et al. \(2020\)](#):

---

An estimand, or quantity of interest, is some summary of parameters or data that somebody is interested in estimating. For example, in the regression model,  $y = a + bx + \epsilon$ , the parameters  $a$  and  $b$  might be of interest.... We use the data to construct estimates of parameters and other quantities of interest.

---

More broadly, [Cunningham \(2021\)](#) defines causal inference as ‘...the leveraging of theory and deep knowledge of institutional details to estimate the impact of events and choices on a given outcome of interest.’ In the previous chapter we discussed gathering data which we observed about the world. In this chapter we are going to be more active. [Cunningham \(2021\)](#) says that experimental data ‘is collected in something akin to a laboratory environment. In a traditional experiment, the researcher participates actively in the process being recorded.’ That is, if we want to use this data then as researchers we have to go out and hunt it, if you like.

### 9.1.3 Randomised sampling

Correlation can be enough in some settings, but in order to be able to make forecasts when things change and the circumstances are slightly different we need to understand causation. The key is the counterfactual - what would have happened in the absence of the treatment. Ideally we could keep everything else constant, randomly divide the world into two groups, and then treat one and not the other. Then we can be pretty confident that any difference between the two groups is due to that treatment. The reason for this is that if we have some population and we randomly select two groups from it, then our two groups (so long as they are both big enough) should have the same characteristics as the population. Randomised controlled trials (RCTs) and A/B testing attempts to get us as close to this ‘gold standard’ as we can hope. RCTs are often described as the ‘gold standard’, for instance by [Athey and Imbens \(2017\)](#). In doing so, we’re not saying that RCTs are perfect, just that

they're generally better than most of the other options. There is plenty that is wrong with RCTs.

Remember that our challenge is ([Gertler et al., 2016](#), p.51-52):

---

...to identify a treatment group and a comparison group that are statistically identical, on average, in the absence of the program. If the two groups are identical, with the sole exception that one group participates in the program and the other does not, then we can be sure that any difference in outcomes must be due to the program. Finding such comparison groups is the crux of any impact evaluation, regardless of what type of program is being evaluated. Simply put, without a comparison group that yields an accurate estimate of the counterfactual, the true impact of a program cannot be established.

---

We might be worried about underlying trends (the issues with before/after comparison), or selection bias (the issue with self-selection), either of which would result in biased estimators. Our solution is randomisation.

To get started, let's generate a simulated dataset and then sample from it. In general, this is a good way to approach problems:

1. generate a simulated dataset;
2. do your analysis on the simulated dataset; and
3. take your analysis to the real dataset.

The reason this is a good approach is that you know roughly what the outcomes should be in step 2, whereas if you go directly to the real dataset then you don't know if unexpected outcomes are likely due to your own analysis errors, or actual results. The first time you generate a simulated dataset it will take a while, but after a bit of practice you'll get good at it. There are also packages that can help, including `DeclareDesign` ([Blair et al., 2019](#)) and `survey` ([Lumley, 2020](#)). Another good reason it's useful to take this approach of simulation is that when you're working in teams the analysis can get started before the data collection and cleaning is completed. That simulation will also help the collection and cleaning team think about tests they should run on their data.

```

library(tidyverse)

set.seed(853)
Construct a population so that 25 per cent of people like blue and 75 per cent
like white.
population <-
 tibble(person = c(1:10000),
 favourite_color = sample(x = c("Blue", "White"),
 size = 10000,
 replace = TRUE,
 prob = c(0.25, 0.75)),
 supports_the_leafs = sample(x = c("Yes", "No"),
 size = 10000,
 replace = TRUE,
 prob = c(0.80, 0.20)),
) %>%
 mutate(in_frame = sample(x = c(0:1),
 size = 10000,
 replace = TRUE)) %>%
 mutate(group = sample(x = c(1:10),
 size = 10000,
 replace = TRUE)) %>%
 mutate(group = ifelse(in_frame == 1, group, NA))

```

We'll get more into this terminology later, but the sampling frame is subset of the population that can actually be sampled, for instance they are listed somewhere. For instance, Lauren Kennedy<sup>4</sup> likes to use the analogy of a city's population, and the phonebook - almost everyone is in there (or at least they used to be), so the population and the sampling frame are almost the same, but they are not.

Now look at the mean for two groups drawn out of the sampling frame.

```

population %>%
 filter(in_frame == 1) %>%
 filter(group %in% c(1, 2)) %>%
 group_by(group, favourite_color) %>%
 count()
#> # A tibble: 4 x 3
#> # Groups: group, favourite_color [4]
#> group favourite_color n
#> <int> <chr> <int>

```

---

<sup>4</sup><https://jazzystats.com/>

```
#> 1 1 Blue 114
#> 2 1 White 420
#> 3 2 Blue 105
#> 4 2 White 369
```

We are probably convinced by looking at it, but to formally test if there is a difference in the two samples, we can use a t-test.

```
library(broom)

population <-
 population %>%
 mutate(color_as_integer = case_when(
 favourite_color == "White" ~ 0,
 favourite_color == "Blue" ~ 1,
 TRUE ~ 999
))

group_1 <-
 population %>%
 filter(group == 1) %>%
 select(color_as_integer) %>%
 as.vector() %>%
 unlist()

group_2 <-
 population %>%
 filter(group == 2) %>%
 select(color_as_integer) %>%
 unlist()

t.test(group_1, group_2)
#>
#> Welch Two Sample t-test
#>
#> data: group_1 and group_2
#> t = -0.30825, df = 988.57, p-value = 0.758
#> alternative hypothesis: true difference in means is not equal to 0
#> 95 percent confidence interval:
#> -0.05919338 0.04312170
#> sample estimates:
#> mean of x mean of y
#> 0.2134831 0.2215190
```

```
We could also use the tidy function in the broom package.
tidy(t.test(group_1, group_2))
#> # A tibble: 1 x 10
#> estimate estimate1 estimate2 statistic p.value parameter conf.low conf.high
#> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 -0.00804 0.213 0.222 -0.308 0.758 989. -0.0592 0.0431
#> # ... with 2 more variables: method <chr>, alternative <chr>
```

If properly done then not only will we get a ‘representative’ share of people with the favourite color blue, but we should also get a representative share of people who support the Maple Leafs. Why should that happen when we haven’t randomised on these variables? Let’s start by looking at our dataset.

```
population %>%
 filter(in_frame == 1) %>%
 filter(group %in% c(1, 2)) %>%
 group_by(group, supports_the_leafs) %>%
 count()
#> # A tibble: 4 x 3
#> group supports_the_leafs n
#> <int> <chr> <int>
#> 1 1 No 102
#> 2 1 Yes 432
#> 3 2 No 81
#> 4 2 Yes 393
```

This is very exciting. We have a representative share on ‘unobservables’ (in this case we do ‘observe’ them - to illustrate the point - but we didn’t select on them). We get this because they were correlated. But it will breakdown in a number of ways that we will discuss. It also assumes large enough groups - if we sampled in Toronto are we likely to get a ‘representative’ share of people who support the Canadiens? What about F.C. Hansa Rostock<sup>5</sup>? If we want to check that the two groups are the same then what can we do? Exactly what we did above - just check if we can identify a difference between the two groups based on observables (we looked at the mean, but we could look at other aspects as well).

#### 9.1.4 ANOVA

---

<sup>5</sup><https://www.fc-hansa.de/>

'I refuse to teach anova.'

Statistics professor who prefers to remain anonymous.

---

Analysis of Variation (ANOVA) was introduced by Fisher while he was working on statistical problems in agriculture. To steal Darren L Dahly's<sup>6</sup> 'favorite joke of all time' (Dahly, 2020):

---

Q: "What's the difference between agricultural and medical research?"

A: "The former isn't conducted by farmers."

---

We need to cover ANOVA because of its importance historically, but in general you probably shouldn't actually use ANOVA day-to-day. There's nothing wrong with it, in the right circumstances, it's more just that it is a hundred years old and the number of modern use-case where it's still your best-bet is pretty small. In any case, typically, the null is that all of the groups are from the same distribution.

We can run ANOVA with the function built into R - `aov()`.

```
just_two_groups <- population %>%
 filter(in_frame == 1) %>%
 filter(group %in% c(1, 2))

aov(group ~ favourite_color,
 data = just_two_groups) %>%
 tidy()
#> # A tibble: 2 x 6
#> term df sumsq meansq statistic p.value
#> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 favourite_color 1 0.0238 0.0238 0.0952 0.758
#> 2 Residuals 1006 251. 0.250 NA NA
```

In this case, we fail to reject the null that the samples are the same. This all said, it's just linear regression. So I'm not sure why it got a fancy name.

---

<sup>6</sup><https://darrendahly.github.io>

```

lm(group ~ favourite_color,
 data = just_two_groups) %>%
 tidy()
#> # A tibble: 2 × 5
#> term estimate std.error statistic p.value
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 (Intercept) 1.48 0.0338 43.8 1.67e-235
#> 2 favourite_colorWhite -0.0118 0.0382 -0.308 7.58e- 1

```

My favourite discussion of ANOVA is [Taback \(2020, Chapter 8\)](#).

### 9.1.5 Treatment and control

If the treated and control groups are the same in all ways and remain that way, then we have internal validity, which is to say that our control will work as a counterfactual and our results can speak to a difference between these groups in that study.

In the words of [Gertler et al. \(2016, p. 71\)](#):

---

Internal validity means that the estimated impact of the program is net of all other potential confounding factors—or, in other words, that the comparison group provides an accurate estimate of the counterfactual, so that we are estimating the true impact of the program.

---

If the group to which we applied our randomisation were representative of the broader population, and the experimental set-up were fairly similar to outside conditions, then we further have external validity. That means that the difference that we find does not just apply in our own experiment, but also in the broader population.

Again, in the words of [Gertler et al. \(2016, p. 73\)](#):

---

External validity means that the evaluation sample accurately represents the population of eligible units. The results of the evaluation can then be generalized to the population of eligible

units. We use random sampling to ensure that the evaluation sample accurately reflects the population of eligible units so that impacts identified in the evaluation sample can be extrapolated to the population.

---

But this means we need randomisation twice. How does this trade-off happen and to what extent does it matter?

As such, we are interested in the effect of being ‘treated’. This may be that we charge different prices (continuous treatment variable), or that we compare different colours on a website (discrete treatment variable, and a staple of A/B testing). If we consider just discrete treatments (so that we can use dummy variables) then need to make sure that all of the groups are otherwise the same. How can we do this? One way is to ignore the treatment variable and to examine all other variables - can you detect a difference between the groups based on any other variables? In the website example, are there a similar number of:

- PC/Mac users?
- Safari/Chrome/Firefox/other users?
- Mobile/desktop users?
- Users from certain locations?

These are all threats to the validity of our claims.

But if done properly, that is if the treatment is truly independent, then we can estimate an ‘average treatment effect’, which in a binary treatment variable setting is:

$$\text{ATE} = \mathbb{E}[y|d = 1] - \mathbb{E}[y|d = 0].$$

That is, the difference between the treated group,  $d = 1$ , and the control group,  $d = 0$ , when measured by the expected value of some outcome variable,  $y$ . So the mean causal effect is simply the difference between the two expectations!

Let’s again get stuck into some code. First we need to generate some data.

```
set.seed(853)
example_data <- tibble(person = c(1:1000),
 treatment = sample(x = 0:1, size = 1000, replace = TRUE))
We want to make the outcome slightly more likely if they were treated than if not.
example_data <-
 example_data %>%
 rowwise() %>%
 mutate(outcome = if_else(treatment == 0,
```

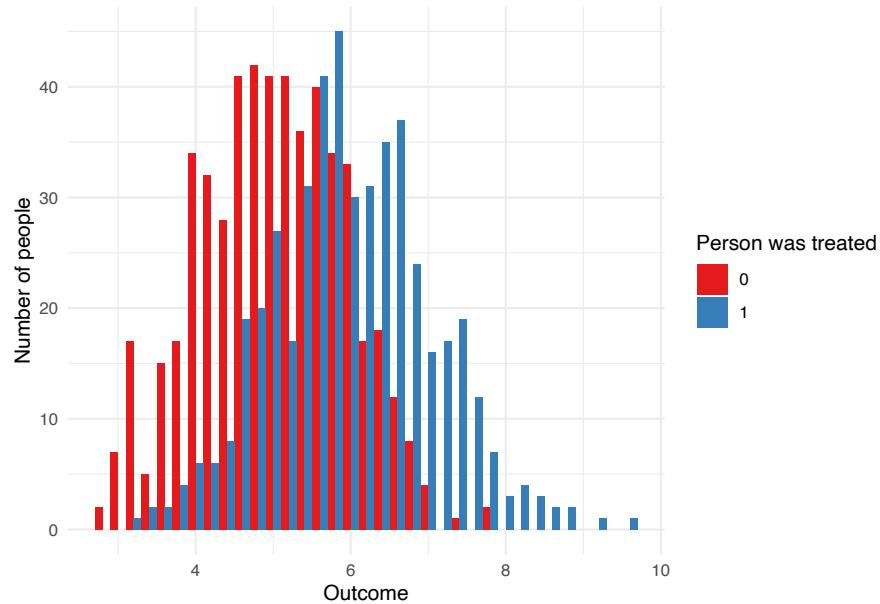
```

 rnorm(n = 1, mean = 5, sd = 1),
 rnorm(n = 1, mean = 6, sd = 1)
)
)

example_data$treatment <- as.factor(example_data$treatment)

example_data %>%
 ggplot(aes(x = outcome,
 fill = treatment)) +
 geom_histogram(position = "dodge",
 binwidth = 0.2) +
 theme_minimal() +
 labs(x = "Outcome",
 y = "Number of people",
 fill = "Person was treated") +
 scale_fill_brewer(palette = "Set1")

```



```

example_regression <- lm(outcome ~ treatment, data = example_data)

tidy(example_regression)
#> # A tibble: 2 x 5

```

```
#> term estimate std.error statistic p.value
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 (Intercept) 5.00 0.0430 116. 0
#> 2 treatment1 1.01 0.0625 16.1 5.14e-52
```

But then reality happens. Your experiment cannot run for too long otherwise people may be treated many times, or become inured to the treatment, but it cannot be too short otherwise you can't measure longer term outcomes. You cannot have a 'representative' sample on every cross-tab, but if not then the treatment and control will be different. Practical difficulties may make it difficult to follow up with certain groups.

Questions to ask (if they haven't been answered already) include:

- How are the participants being selected into the frame for consideration?
- How are they being selected for treatment? We would hope this is a lottery, but this term is applied to a variety of situations. Additionally, early 'success' can lead to pressure to treat everyone.
- How is treatment being assessed?
- To what extent is random allocation ethical and fair? Some argue that shortages mean it is reasonable to randomly allocate, but that may depend on how linear the benefits are. It may also be difficult to establish boundaries. If we only want to include people in Ontario then that may be clear, but what about 'students' in Ontario - who is a student, and who is making the decision?

Bias and other issues are not the end of the world. But you need to think about it carefully. In the famous example, Abraham Wald was given data on the planes that came back to Britain after being shot at in WW2. The question is where to place the armour. One option is to place it over the bullet holes. Wald recognised that there is a selection effect here - these are the planes that made it back - they didn't need the armour, but instead we should put the armour where there were no bullet holes.

To consider an example that may be closer to home - how would the results of a survey differ if I only asked students who completed this course what was difficult about it and not those who dropped out? While, as Dan suggests, we should work to try to make the dataset as good as possible, it may be possible to use the model to control for some of the bias. If there is a variable that is correlated with say, attrition, then we can add it to the model. Either by itself, or as an interaction.

What if there is a correlation between the individuals? For instance, what if there were some 'hidden variable' that we didn't know about, such as province, and it turned out that people from the same province were similar? In that case we could use 'wider' standard errors.

But a better way to deal with this may be to change the experiment. For instance, we discussed stratified sampling - perhaps we should stratify by province? How would we implement this? And of course, these days we'd not really use a 100-year-old method but would instead use Bayes-based approaches.

---

## 9.2 Case study - Fisher's tea party

Fisher (see note above) introduced a, now, famous example of an experiment designed to see if a person can distinguish between a cup of tea when the milk was added first, or last.<sup>7</sup>

From Fisher (1935, p.13):

---

A lady declares that by tasting a cup of tea made with milk she can discriminate whether the milk or the tea infusion was first added to the cup. We will consider the problem of designing an experiment by means of which this assertion can be tested.

---

Fisher continues:

---

Our experiment consists in mixing eight cups of tea, four in one way and four in the other, and presenting them to the subject for judgment in a random order. The subject has been told in advance of what the test will consist, namely that she will be asked to taste eight cups, that these shall be four of each kind, and that they shall be presented to her in a random order, that is in an order not determined arbitrarily by human choice, but by the actual manipulation of the physical apparatus used in games of chance, cards, dice, roulettes, etc., or, more expeditiously, from a published collection of random sampling-numbers purporting to give the actual results of such manipulation. Her task is to divide the 8 cups into two sets of 4, agreeing, if possible, with the treatments received.

---

<sup>7</sup>I'm personally very attached to this example as this issue also matters a lot to my father



**FIGURE 9.1:** Afternoon Tea Party (1890–1891), by Mary Cassatt (American, 1844–1926), as downloaded from <https://artvee.com/dl/afternoon-tea-party>.

---

To summarize, the set-up is:

- Eight randomly ordered cups of tea.
- Four had tea put in first.
- Four had milk put in first.
- The person has to choose the four that are the same.
- The person knows it's an experiment.

We'll now try this experiment. So brew some tea, grab eight cups, and pour eight cups of tea for a friend that you're isolating with<sup>8</sup> - four where you put the milk in first and four where you put the milk in last. Make sure you use the same amount of tea and milk in each! Don't forget to randomise the order, possibly even using the following code:

```
sample(c(1:8), size = 8, replace = FALSE)
#> [1] 3 7 6 4 1 8 2 5
```

Then have your friend guess which four you put milk in first and which four you put milk in last!

To decide if the person's choices were likely to have occurred at random or not, we need to think about the probability of this happening by chance. First count the number of successes out of the four that were chosen. Fisher (1935, p.14) claims there are:  $\binom{8}{4} = \frac{8!}{4!(8-4)!} = 70$  possible outcomes.

By chance, there are two ways for the person to be perfectly correct (because we are only asking them to be grouped): correctly identify all the ones that were milk-first (one outcome out of 70) or correctly identify all the ones that were tea-first (one outcome out of 70), so the chance of that is  $2/70 \approx 0.028$ . Now, as Fisher (1935, p.15) says,

---

'[i]t is open to the experimenter to be more or less exacting in respect of the smallness of the probability he would require before he would be willing to admit that his observations have demonstrated a positive result'.

---

You need to decide what evidence it takes for you to be convinced. If there's

---

<sup>8</sup>For posterity, 2020 was quite a year.

no possible evidence that will dissuade you from your view (that there is no difference between milk-first and tea-first) then what is the point of doing an experiment? In any case, if the null is that they can't distinguish, but they correctly separate them all, then at the five-per-cent level, we reject the null.

What if they miss one? Similarly, by chance there are 16 ways for a person to be 'off-by-one'. Either they think there was one that was milk-first when it was tea-first - there are,  $\binom{4}{1}$ , four ways this could happen - or they think there was one that was tea-first when it was milk-first - again, there are,  $\binom{4}{1}$ , four ways this could happen. But these outcomes are independent, so the probability is  $\frac{4 \times 4}{70} \approx 0.228$ . And so on. So, we fail to reject the null.

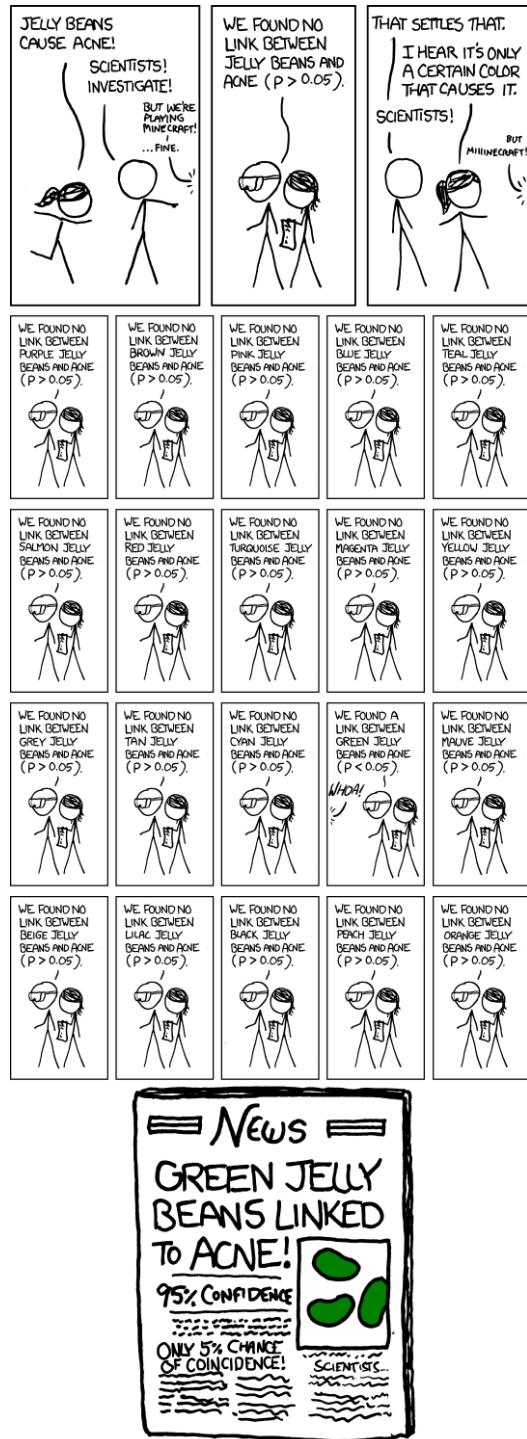
Finally, an aside on this magical '5 per cent'. Fisher himself describes this as merely 'usual and convenient' (Fisher, 1935, p.15). Fisher (1935, p.16) continues:

---

In order to assert that a natural phenomenon is experimentally demonstrable we need, not an isolated record, but a reliable method of procedure. In relation to the test of significance, we may say that a phenomenon is experimentally demonstrable when we know how to conduct an experiment which will rarely fail to give us a statistically significant result.

---

At the start of these notes, I said that Fisher held views that we would consider reprehensible today. My guess is, were he around today, he would think our use of p-values as discrediting. Do not just go searching for meaning in constellations of stars. Thoroughly interrogate your data and think precisely about the statistical methods you are applying. For conclusions that you want to hold up in the long-run, aim to use as simple, and as understandable, statistical methods as you can. Ensure that you can explain and justify your statistical decisions without recourse to astrology.



Source: <https://xkcd.com/882/>



**FIGURE 9.2:** 'The triumph of wisdom over fortune' by Otto van Veen (Flemish, 1556 - 1629), as downloaded from <https://artvee.com/dl/the-triumph-of-wisdom-over-fortune>.

---

### 9.3 Case study - Tuskegee Syphilis Study

The Tuskegee Syphilis Study is an infamous medical trial in which Black Americans with syphilis (and a ‘control group’ without) were not given appropriate treatment, nor even told they had syphilis, well after standard syphilis treatments were established in the mid-1940s (Alsan and Wanamaker, 2018). The study began in 1932 when poor Black Americans in the South were identified and offered compensation including ‘hot meals, the guise of treatment, and burial payments’ (Alsan and Wanamaker, 2018). The men were not treated for syphilis. Further, and this is almost unbelievable, some of the men were drafted, told they had syphilis, and ordered to get treatment. This treatment was blocked. By the time the study was stopped, ‘the majority of the study’s victims were deceased, many from syphilis-related causes.’ (Alsan and Wanamaker, 2018).

The study continued through to 1972, only stopping when it was leaked and published in newspapers. In response the US established requirements for Institutional Review Boards and President Clinton made a formal apology in 1997. Brandt (1978) as quoted by Alsan and Wanamaker (2018) says ‘“In retrospect the Tuskegee Study revealed more about the pathology of racism

than the pathology of syphilis; more about the nature of scientific inquiry than the nature of the disease process.... The degree of deception and the damages have been severely underestimated.”<sup>9</sup>

On the Tuskegee Syphilis Study Professor Monica Alexander<sup>9</sup> says:

---

While it may be illegal to do this exact research these days, it doesn't mean that unethical research doesn't still happen, and we see it all the time in ML and health. Just because you can't explicitly discriminate when you design experiments, doesn't mean you can't implicitly discriminate.

---

For an example of this, start with Obermeyer et al. (2019):

---

Health systems rely on commercial prediction algorithms to identify and help patients with complex health needs. We show that a widely used algorithm, typical of this industry-wide approach and affecting millions of patients, exhibits significant racial bias: At a given risk score, Black patients are considerably sicker than White patients, as evidenced by signs of uncontrolled illnesses. Remedyng this disparity would increase the percentage of Black patients receiving additional help from 17.7 to 46.5%. The bias arises because the algorithm predicts health care costs rather than illness, but unequal access to care means that we spend less money caring for Black patients than for White patients. Thus, despite health care cost appearing to be an effective proxy for health by some measures of predictive accuracy, large racial biases arise. We suggest that the choice of convenient, seemingly effective proxies for ground truth can be an important source of algorithmic bias in many contexts.

---

<sup>9</sup><https://www.monicaalexander.com>

## 9.4 Sampling and survey essentials

### 9.4.1 Introduction

Statistics is the cold, hard, core of using data. Statisticians have spent considerable time and effort thinking about the properties that various samples of data will have and how they enable us to speak to implications for the broader population.

Let's say that we have some data. For instance, a particular toddler goes to sleep at 6:00pm every night. We might be interested to know whether that bedtime is common more generally among all toddlers, or if we have an unusual toddler. We only have one toddler so our ability to use his bed time to speak about all toddlers is limited. But what about if we talk to our friends who also have toddlers? How many friends, and friends of friends, do we have to ask because we can begin to feel comfortable speaking about some underlying truth of toddler bedtime?

In the wonderful phrase of [Wu and Thompson \(2020, p. 3\)](#) '[s]tatistics is the science of how to collect and analyze data, and draw statements and conclusions about unknown populations. The term population usually refers to a real or hypothetical set of units with characteristics and attributes which can be modelled by random variables and their respective probability distributions.' In my own much less wonderful phrasing, 'statistics involves having some data and trying to say something sensible about it'. I mean, it's really up to you which one you want to go with.

In the case of surveys, our population is a finite set of  $N$  labels: 'person 1', 'person 2', 'person 3', ..., 'person  $N$ '. It is important here to recognise that there is a difference between the population of interest to a survey and a population in the sense that it is used when we talk of limits and similar infinity concepts in statistics. For instance, from time to time, you hear people who work with census data say that they don't need to worry about confidence intervals because they have the whole population of the country. Nothing could be further from the truth.

[Wu and Thompson \(2020, p. 4\)](#) have a lovely example of the ambiguity that surrounds the definition of a population. Let's consider the population of voters. In Canada that means anyone who is 18 or older. Fine. But what if we are interested in consumers - what is the definition of hipsters? I regularly eat avocado toast, (+1), but I've never had bullet coffee (-1). Am I in the population or not?

More things are formally defined than you may realise. For instance, the idea of a rural area is precisely defined. A property is either in a rural area or not. But then we come to the lovely example of [Wu and Thompson \(2020,](#)

p. 4) when it comes to whether someone is a smoker. If a 15 year old has had 100 cigarettes then it's pretty clear that we need to treat them differently than if they have had none. But if a 100 year old has had 100 cigarettes then we consider them to have none. That's fine, but what is the age at which this changes? Further, think about how this changes over time. At one point, parents used to be worried if children had more than two hours of screen time, now those same children (and possibly even the parents) regularly likely spend more than eight hours in front of a screen if they work in an office job.

So we come to some critical terminology:

1. Population: 'The set of all units covered by the main objective of the study' [Wu and Thompson \(2020, p. 5\)](#).
2. Frame: 'Lists of sampling units' [Wu and Thompson \(2020, p. 9\)](#) where sampling units are either the observational units themselves or the clusters.
3. Sample: Those who complete and return the survey.

To be a little more concrete about this, consider that we are trying to conduct a survey about the attitudes Australians who live in Toronto. So the target population is all Australians who live in Toronto, the frame might be all those Australians who live in Toronto who use Facebook, because we are going to use Facebook to choose who to sample. And then finally, if we take that Facebook list of all Australians living in Canada and we gave each one a chance at being surveyed then that would be our sampled population, but if we just picked the ones that I know then it would just be Dan, Monica, and Liza (from New Zealand but we'll claim her because that's a thing that Australians do).

In that example the target population and the frame will be different because not all Australians who live in Toronto are on Facebook. Similarly, if not everyone that we gave the survey to actually completed the survey then the sample and the frame would be different.

Having identified a population of interest and a frame (i.e. a list that gets the closest to that population) At this point we distinguish between probability and non-probability sampling.

With probability sampling, every member of the frame has some chance of being sampled. Consider the example of the Australian Election Study - they get a list of all the addresses in Australia, and then randomly choose some to send letters to. The 'randomista' and RCT revolution that we discuss later, is needed because of a lack of probability sampling, but when it exists it plays a role here. Importantly it ensures that we are clear about the role of uncertainty ([Wu and Thompson, 2020, p. 11](#)). The trade-off is that it is expensive and difficult. Note that each unit in the frame doesn't have to have the same probability necessarily, it just needs to be determined by a probability measure.

In contrast, with non-probability sampling we focus on populations that are ‘readily available’ or convenient, satisfy certain quotas, based on judgement, or those that volunteer. The difference between probability and non-probability sampling is that of degree - we typically cannot force someone to take our survey, and hence, there is almost also an aspect of volunteering.

While acknowledging that it is a spectrum, most of statistics was developed based on probability sampling. But much of modern sampling is done using non-probability sampling. In particular, a common approach is to have a bunch of Facebook ads trying to recruit a panel of people in exchange for compensation. This panel is then the group that is sent various surveys as necessary. But think for a moment about the implications of this - what type of people are likely to respond to such an ad? I don’t know who Canada’s richest person is, but are they likely to be in this panel? Is your grandmother likely to respond to that ad? What about you - do you even use Facebook?

In some cases it is possible to do a census. Nation-states typically do one every five to ten years. But there is a reason that it is only nation states that do them - they are expensive, time-consuming, and surprisingly, they are sometimes not as accurate as we may hope because of how general they need to be. Hence, the role of surveys. Note, however that censuses will typically have many of the same concerns.

When we consider our population, it will typically have some ordering. This may be as simple as a country having states/provinces. We consider a stratified structure to be one in which we can divide the population into mutually exclusive and collectively exhaustive sub-populations, or strata. Examples of strata in [Wu and Thompson \(2020, p. 8\)](#) include provinces, federal electoral districts, or health regions. But strata need not be geographic, and it may be possible to use different majors. We use stratification to help with the efficiency of sampling or with the balance of the survey. For instance, if we surveyed provinces in proportion to their population, then even a survey of 10,000 responses would only expect to have 10 responses from the Yukon.

The other word that is used that takes advantage of the ordering of some population is clusters. Again, these are collectively exhaustive and mutually exclusive. Again, they may be geographically based, but need not be. The difference between stratified sampling and cluster sampling, is that ‘under stratified sampling, sample data are collected from every stratum, (whereas) under cluster sampling, only a portion of the clusters has members in the final sample’ [Wu and Thompson \(2020, p. 8\)](#). That all said, this difference can become less clear in practice, especially *ex post* - what if you stratify then randomly sample within that strata, but no one is selected - but in terms of intention the difference is clear.

We now turn to the first of our claims, which is that if we have a perfect frame and no non-response, then our sample results will match that of the

population. We'd of course be very worried if that weren't the case, but it's nice to have it stated. We establish some type of population mean for the study variable,  $\mu_y$ , and population means for the auxiliary variables  $\mu_x$ , which could be things like age, gender, etc. Remembering that when we do this in the real world, we may have many study variables, and indeed, some overlap. If a variable is an indicator then in this set-up all we have to do is to work out the proportion in order to estimate it, which is  $P$ . And finally, we get a rule of thumb for large samples whereby the variance in this binary and perfect setting becomes  $\sigma_y^2 = P/(1 - P)$  (Wu and Thompson, 2020, p. 11).

Finally, we conclude with the steps that you should consider. These are all critical. Strong reports would grapple with all of these.

#### 9.4.2 Simple random sampling

TBD

#### 9.4.3 Stratified and cluster sampling

TBD

#### 9.4.4 Snowball sampling and confidant methods

TBD

---

### 9.5 Case study - The Oregon Health Insurance Experiment

The Oregon Health Insurance Experiment involved 74,922 adults in Oregon from 2008 to 2010. The opportunity to apply for health insurance was randomly allocated and then health and earnings evaluated. It was found that (Finkelstein et al., 2012):

---

In the year after random assignment, the treatment group selected by the lottery was about 25 percentage points more likely to have insurance than the control group that was not selected. We find that in this first year, the treatment group had substantively and statistically significantly higher health care utilization (including primary and preventive care as well as hospitalizations), lower out-of-pocket medical expenditures and medical

debt (including fewer bills sent to collection), and better self-reported physical and mental health than the control group.

---

A lottery was used to determine which of the 89,824 individuals who signed up would be allowed to apply for Medicaid. This random allocation of insurance allowed the researchers to understand the effect of health insurance. It's not usually possible to compare those with and without insurance because the type of people that sign up to get health insurance differ to those who don't - that decision is 'confounded' with other variables. They use administrative data, such as hospital discharge data, credit reports that were matched to 68.5 per cent of lottery participants, and mortality records, which will be uncommon. Interestingly this collection of data is actually fairly restrained and so they included a survey conducted via mail.

Turning to external validity, the authors restrain themselves and say ([Finkelstein et al., 2012](#)):

---

Our estimates of the impact of public health insurance apply to able-bodied uninsured adults below 100 percent of poverty who express interest in insurance coverage. This is a population of considerable policy interest.

---

A lottery was used to allocate 10,000 places in the state-run Medicaid. A lottery was judged fair because 'the state (correctly) anticipated that the demand for the program among eligible individuals would far exceed the 10,000 available new enrollment slots' ([Finkelstein et al., 2012](#)). People had a month to sign up to enter the draw. The draws were conducted over a six-month period and those who were selected had the opportunity to sign up. 35,169 individuals were selected (the household of those who actually won the draw was given the opportunity) but only 30 per cent of them completed the paperwork and were eligible (typically they earned too much). The insurance lasted indefinitely.

The model they consider is ([Finkelstein et al., 2012](#)):

$$y_{ihj} = \beta_0 + \beta_1 \text{Lottery} + X_{ih}\beta + 2 + V_{ih}\beta_3 + \epsilon_{ihj} \quad (9.1)$$

Equation (9.1) explains various  $j$  outcomes (such as health) for an individual  $i$  in household  $h$  as a function of an indicator variable as to whether household  $h$

was selected by the lottery. Hence, '(t)he coefficient on Lottery,  $\beta_1$ , is the main coefficient of interest, and gives the average difference in (adjusted) means between the treatment group (the lottery winners) and the control group (those not selected by the lottery).'

To complete the specification of Equation (9.1),  $X_{ih}$  is a set of variables that are correlated with the probability of being treated. These adjust for that impact to a certain extent. An example of that is the number of individuals in a household. And finally,  $V_{ih}$  is a set of variables that are not correlated with the lottery. These variables include demographics, hospital discharge and lottery draw.

There is a wide range of literature related to this intervention. More papers are available here<sup>10</sup>.

---

## 9.6 Case study - Student Coaching: How Far Can Technology Go?

There is a general concern about students dropping out of university before they finish their degree. If you work one-on-one with a student then this addresses the issue. But that doesn't scale. The point of this experiment was to see if technology-based options could be more efficient. The focus was the University of Toronto, and in particular first-year economics courses in Fall 2015.

The intervention was administered to students as part of an economics class. Students received 2 per cent of their grade for completing the exercise. The specific exercise depended on the group of the student. The intervention involved three treatments as well as a control group that was just given a Big Five personality traits test. Additional information that was obtained included 'the highest level of education obtained by students' parents, the amount of education they expect to obtain, whether they are first-year or international students, and their work and study time plans for the upcoming year.' (Oreopoulos and Petronijevic, 2018, p. 6).

The treatments were (Oreopoulos and Petronijevic, 2018, p. 4):

1. '[A] one-time, online exercise completed during the first two weeks of class in the fall'. This exercise was 'designed to get them thinking about the future they envision and the steps they could take in the upcoming year at U of T to help make that future a reality.'

---

<sup>10</sup><https://www.povertyactionlab.org/evaluation/oregon-health-insurance-experiment-united-states>

They were told that the exercise was designed for their benefit and to take their time while completing it. The online module lasted approximately 60 to 90 minutes and led students through a series of writing exercises in which they wrote about their ideal futures, both at work and at home, what they would like to accomplish in the current year at U of T, how they intend on following certain study strategies to meet their goals, and whether they want to get involved with extracurricular activities at the university' (Oreopoulos and Petronijevic, 2018, p. 6).

2. '[T]he online intervention plus text and email messaging throughout the full academic year'. This involved the students being given 'the opportunity to provide their phone numbers and participate in a text and email messaging campaign lasting throughout both the fall semester in 2015 and the winter semester in 2016' (Oreopoulos and Petronijevic, 2018, p. 8). All students in this group got the emails, but only those that provided phone numbers got the messages. They were able to opt out, but 'few chose to do so' (Oreopoulos and Petronijevic, 2018, p. 8). This was a two-way interaction in which students could ask questions. Some asked for the 'locations of certain facilities on campus or how to stay on residence during the holiday break, while others said they need help with English skills or specific courses. Some students expressed relatively deep emotions, such as feeling anxious about family pressure to succeed in school or from doing poorly on an evaluation' (Oreopoulos and Petronijevic, 2018, p. 9). A response was usually given within an hour.
3. '[T]he online intervention plus one-on-one coaching in which students are assigned to upper-year undergraduate coaches'. 'Coaches were available to meet with students to answer any questions via Skype, phone, or in person, and would send their students regular text and email messages of advice, encouragement, and motivation, much like the You@Uoft<sup>11</sup> program described above. In contrast to the messaging program, however, coaches were instructed to be proactive and regularly monitor their students' progress. Whereas the You@Uoft<sup>12</sup> program attempts to "nudge" students in the right direction with academic advice, coaches play a greater "support" role, sensitively guiding students through problems.' (Oreopoulos and Petronijevic, 2018, p. 11). This coaching program was only available at UTM. 'Our coaching treatment group was established by randomly drawing twenty-four students from the group of students that were randomly assigned into the text message campaign treatment. At the conclusion of the online exercise, instead of being

---

<sup>11</sup><mailto:You@Uoft>

<sup>12</sup><mailto:You@Uoft>

invited to provide a phone number for the purpose of receiving text messages, these twenty-four students were given the opportunity to participate in a pilot coaching program. A total of seventeen students agreed to participate in the coaching program, while seven students declined.<sup>13</sup>

---

Our coaching treatment group was established by randomly drawing twenty-four students from the group of students that were randomly assigned into the text message campaign treatment. At the conclusion of the online exercise, instead of being invited to provide a phone number for the purpose of receiving text messages, these twenty-four students were given the opportunity to participate in a pilot coaching program. A total of seventeen students agreed to participate in the coaching program, while seven students declined.

(Oreopoulos and Petronijevic, 2018, p. 14)

---

The model they consider is (Oreopoulos and Petronijevic, 2018, p. 15):

$$y_{ij} = \alpha + \beta_1 \text{Online}_i + \beta_2 \text{Text}_i + \beta_3 \text{Coach}_i + \delta_j + \mu \text{First year}_i + \epsilon_{ij} \quad (9.2)$$

Equation (9.2) explains the outcome of student  $i$  at campus  $j$  based on ‘indicators for each of the three treatment exercises students were given, campus fixed effects, and a first-year student indicator.’ The main parameters of interest are  $\beta_1$ ,  $\beta_2$  and  $\beta_3$ . The main outcomes were course grades, GPA, credits earned and failed.

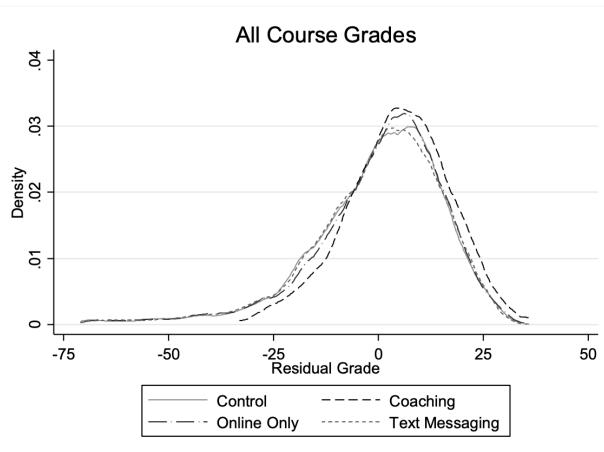
It was found, that the one-on-one coaching ‘increased grades by approximately 5 percentage points’, while the other treatments had ‘had no detectable impact’. One set of results are summarised in Figure 9.3.

The results are important not only in a teaching context, but also for businesses hoping to retain customers. More papers are available here<sup>13</sup>.

---

<sup>13</sup><https://www.povertyactionlab.org/evaluation/student-coaching-how-far-can-technology-go>

Figure 1  
Grade Distributions Across All Courses by Treatment Status



This figure presents residual grade distributions using grade outcomes from all (full-year, winter, and fall) courses. To construct the figure, we stack course grade outcomes for all students from all courses, regress course grades on campus and first-year fixed effects, and obtain the residuals from this regression. The figure shows the density of these residuals for each of the three treatment groups and the control group.

**FIGURE 9.3:** Example of the results of the intervention.

## 9.7 Case study - Civic Honesty Around The Globe

Trust isn't something that we think regularly about, but it's actually fairly fundamental to most interactions, both economic and personal. For instance, many of us get paid after we do some work - we're trusting our employer will make good; and vice versa - if you get paid in advance then they are trusting you. In a strictly naive, one-shot, transaction-cost-less world, this doesn't make sense. If you get paid in advance, the incentive is for you to take the money and run in the last pay period before you quit, and through backward induction everything falls apart. Of course, we don't live in such a world. For one thing there are transaction costs, for another generally we have repeated interactions, and finally, in my experience, the world usually ends up being fairly small.

Understanding the extent of honesty in different countries may help us to explain economic development and other aspects of interest such as tax compliance, but it's fairly hard to measure. We can't really ask people how honest they are - wouldn't the liars lie, resulting in a lemons problem ([Akerlof, 1978](#))? To get around, this [Cohn et al. \(2019a\)](#) conduct an experiment in 355 cities across 40 countries where they 'turn in' either: a wallet with the local equivalent of US\$13.45 in it, or no money. They are interested in whether the 'recipient' attempts to return the wallet. They find that '[in virtually all countries, citizens were more likely to return wallets that contained more money' ([Cohn et al., 2019a](#), p. 1).

The set-up of the experiment is fascinating. They 'turn in' 17,303 wallets to various institutions including: '(i) banks; (ii) theaters, museums, or other cultural establishments; (iii) post offices; (iv) hotels; and (v) police stations, courts of law, or other public offices' ([Cohn et al., 2019a](#), p. 1). These institutions were roughly equally sampled, although banks were slightly over sampled and post offices were slightly under-sampled. The importance of such institutions in the economy is generally well-accepted ([Acemoglu et al., 2001](#)) and they are common across most countries. Importantly for the experiment, they 'typically have a public reception area where we could perform the drop-offs' ([Cohn et al., 2019a](#), p. 1).

The way the experiment worked is that a research assistant turned the wallet in to an employee at a counter in the public reception area, saying 'Hi, I found this [showing the wallet] on the street just around the corner. [Place wallet on counter.] Somebody must have lost it. I'm in a hurry and have to go. Can you please take care of it?' ([Cohn et al., 2019a](#), p. 2). The outcome of interest is whether an email is sent to the unique address on a business card in the wallet within 100 days. The research assistant had to note various features of the setting, including features such as the gender, age-group, and busyness of the 'recipient'.

The wallets were transparent, and the business card had a name and email contact details. It also had a key and a grocery list (Figure 9.4).

The grocery list was an attempt to convince the 'recipient' that the 'owner' was a local. Language and currency were adapted to local conditions. The key is only useful to the 'owner', not to the 'recipient' of the wallet and was included to test for altruistic concerns.

The primary treatment in the experiment is whether the wallet contained money or not. The key outcome was whether the wallet was attempted to be returned or not. It was found that '[t]he median response time was roughly 26 minutes across all countries, and about 88% of emails arrived within 24 hours' ([Cohn et al., 2019b](#), p. 10). If an email was received, then 3 hours later a response was sent, saying that the owner had left town, the contents were

Fig. S1. Example lost wallet



Example of a wallet used in our field experiments. All wallets belonged to a male software developer with country-specific names (see Table S1 for the complete list of names). We placed the business cards in the wallets so that this information was visible to all participants. The wallet dimensions were 93mm x 59mm x 5mm and it weighed approximately 24 grams in the NoMoney condition.

**FIGURE 9.4:** Example of the wallet.

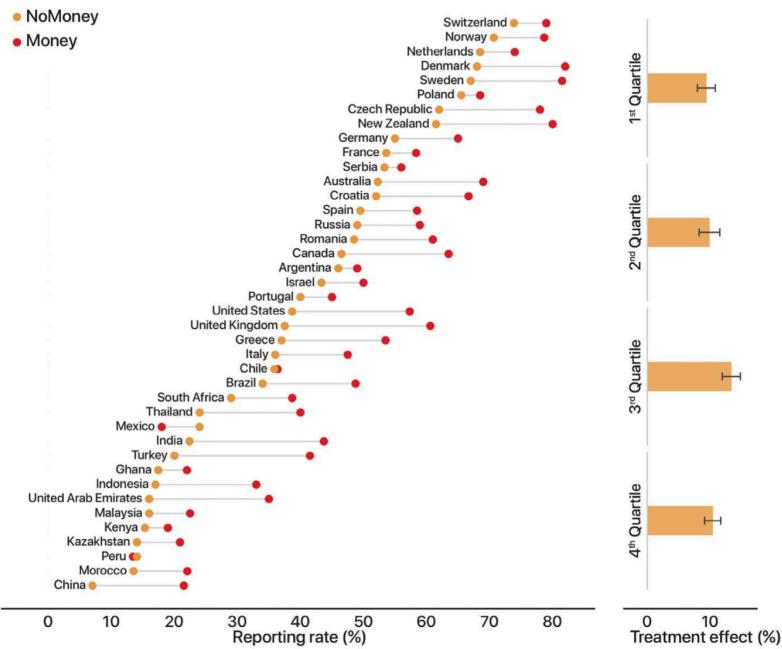
unimportant to them and that they could keep it or donate it to charity (Cohn et al., 2019b, p. 9).

Considerable differences were found between countries (Figure 9.5).

Figure 9.5 shows that in almost all countries wallets with money were more likely to be returned than wallets without. The authors further conducted the experiment with the equivalent of US\$94.15 in three countries - Poland, the UK, and the US - and found that reporting rates further increased. In those same three countries further tests were done comparing the situation when the wallet always contained money, but the presence of the key was varied. The wallet was slightly more likely to be reported when there was a key.

The full set of 40 countries were chosen based on having enough cities with populations of at least 100,000, as well as the ability for the research assistants to safely visit and withdraw cash. The cities were chosen starting with the largest ones and there were usually 400 observations in each country (Cohn et al., 2019b, p. 5). Real-world concerns affected the specifics of the experiment. For instance, in ‘...India, we made a last minute change by replacing Chennai with Coimbatore due to severe flooding that took place in February 2015. In Kenya we did not carry out data collection in the last city visited (Malindi) because the research assistant was arrested and interrogated by the military police for suspicious activity’ (Cohn et al., 2019b, p. 5).

In addition to the experiments, Cohn et al. (2019a) conducted surveys that allowed them to understand some reasons for their findings. It also allowed them to be specific about the respondents. The survey involved 2,525 respondents (829 in the UK, 809 in Poland, and 887 in the US) (Cohn et al., 2019b, p. 36). ‘To qualify for participation, individuals had to pass a simple attention check and meet the demographic quotas (based on age, gender, and residence)



**Fig. 1. Share of wallets reported in the NoMoney and Money conditions, by country. (Left)** Share of wallets reported in NoMoney (US\$0) and Money (US\$13.45) conditions, by country. The amount of money in the wallet is adjusted according to each country's purchasing power. **(Right)** Average difference between Money and NoMoney conditions across quartiles based on absolute reporting rates in the NoMoney condition. Error bars represent standard errors of the mean.

**FIGURE 9.5:** Key finding as to wallet return rates.

set by Qualtrics to construct the representative samples. Participants received a flat payment of US \$4.00 for their participation' (Cohn et al., 2019b, p. 36). The participants were given one of the scenarios and then asked to answer questions.

Annoyingly the authors don't explicitly specify the estimating equation. However they do say that important covariates about the 'recipient' include: gender, age-group, busyness, whether they were local, spoke English, understood the situation, friendliness, presence of: a computer, co-workers, bystanders, security cameras, security guards. Important covariates at a country-level include: country GDP, soil fertility, latitude, distance to water, temperature and its volatility, precipitation and its volatility, elevation, terrain roughness, pathogens, language features such as: pronouns, politeness, future time; share of protestants; family ties; state history; years of democracy; executive constraints; judicial independence; constitutional review; electoral rule; and primary school enrollment in 1920.

The code or data for the paper are available: [Cohn \(2019\)](#).

---

## 9.8 A/B testing

### 9.8.1 Introduction

---

Large companies, particularly tech companies, have developed incredibly sophisticated infrastructure for running complex experiments. In the tech industry, these experiments are often called A/B tests because they compare the effectiveness of two treatments: A and B. Such experiments are frequently run for things like increasing click-through rates on ads, but the same experimental infrastructure can also be used for research that advances scientific understanding.

[Salganik \(2018, p. 185\)](#).

---

The past decade has probably seen the most experiments ever run by several orders of magnitude with the extensive use of A/B testing on websites. Every time you are online you are probably subject to tens, hundreds, or potentially thousands, of different A/B tests. If you use apps like TikTok then this could run to the tens of thousands. While, at their heart, they are still just surveys that result in data that need to be analysed, they have several interesting features, which we will discuss.

The opening example of [Kohavi et al. \(2020, p. 3\)](#) is a particularly nice illustration.

---

In 2012, an employee working on Bing, Microsoft's search engine, suggested changing how ad headlines display ([Kohavi and Thomke 2017](#)). The idea was to lengthen the title line of ads by combining it with the text from the first line below the title, as shown in Figure 1.1.

Nobody thought this simple change, among the hundreds sug-

gested, would be the best revenue-generating idea in Bing's history!

The feature was prioritized low and languished in the backlog for more than six months until a software developer decided to try the change, given how easy it was to code. He implemented the idea and began evaluating the idea on real users, randomly showing some of them the new title layout and others the old one. User interactions with the website were recorded, including ad clicks and the revenue generated from them. This is an example of an A/B test, the simplest type of controlled experiment that compares two variants: A and B, or a Control and a Treatment.

A few hours after starting the test, a revenue-too-high alert triggered, indicating that something was wrong with the experiment. The Treatment, that is, the new title layout, was generating too much money from ads. Such "too good to be true" alerts are very useful, as they usually indicate a serious bug, such as cases where revenue was logged twice (double billing) or where only ads displayed, and the rest of the web page was broken.

For this experiment, however, the revenue increase was valid. Bing's revenue increased by a whopping 12%, which at the time translated to over \$100M annually in the US alone, without significantly hurting key user-experience metrics. The experiment was replicated multiple times over a long period.

The example typifies several key themes in online controlled experiments:

- It is hard to assess the value of an idea. In this case, a simple change worth over \$100M/year was delayed for months.
- Small changes can have a big impact. A \$100M/year return-on-investment (ROI) on a few days' work for one engineer is about as extreme as it gets.
- Experiments with big impact are rare. Bing runs over 10,000 experiments a year, but simple features resulting in such a big improvement happen only once every few years.
- The overhead of running an experiment must be small. Bing's engineers had access to ExP, Microsoft's experimentation system, which made it easy to scientifically evaluate the idea.
- The overall evaluation criterion (OEC, described more later in this chapter) must be clear. In this case, revenue was a key component of the OEC, but revenue alone is insufficient as an OEC. It could lead to plastering the web site with ads, which is known to hurt the user experience. Bing uses an OEC that weighs revenue against user-experience metrics, including Sessions per user (are users abandoning or increasing engagement) and several other com-

ponents. The key point is that user-experience metrics did not significantly degrade even though revenue increased dramatically.

---

In these notes, I'm going to use A/B testing to strictly refer to the situation in which we're dealing with a tech firm, and some type of change in code. If we are dealing with the physical world then we'll stick with RCTs.

I'm usually fairly dismissive of the CS folks who adopt different language for concepts that have been around for a long time. However, in the case of A/B testing I think that it's possibly justified. There is something different about doing tens of thousands of small experiments all the time, compared with our normal RCT set-up of one experiment conducted over months. And finally, if you don't work in a tech firm, then don't discount the difficulty of shifting to an experimental set-up. You may think that it's easy to go to a workplace and say 'hey, let's test stuff before we spend thousands/millions of dollars'. You'd be wrong. In my opinion, the hardest part of A/B testing isn't the science, it's the politics.

### 9.8.2 Delivery

Drawing on [Kohavi et al. \(2020, p. 153-161\)](#), we first consider how we will be delivering the A/B test. In the case of a RCT it's fairly obvious how we deliver it - for instance, make a person come to a doctor's clinic and inject them with a drug or a placebo. In the case of A/B testing, it's less obvious - do you run it 'server-side' or 'client-side'? What this means is, do you just change the website - 'server side', or do you change an app - 'client side'. This may seem like a silly issue, but it affects: 1) release; and 2) data transmission.

In the case of the effect on release, it's easy and normal to update a website all the time, so small changes can be easily implemented in the case of server-side. However, in the case of client-side, let's say an app, it's likely a much bigger deal.

1. It needs to get through an app store (a bigger or lesser deal depending on which one).
2. It need to go through a release cycle (a bigger or lesser deal depending on the specifics of the company and how it ships).
3. Users have the opportunity to not upgrade. Are they likely different to those that do upgrade? (Yes.)

Now, in the case of the effect on data transmission, again server-side is less of a big deal - you kind of get the data as part of the user interacting. But in the case of client-side - it's not necessarily the case that the user will

have the internet at the time they're using your application, and if they do, they may have limitations on the data uploads. The phone may limit data transmission depending on its effect on battery, CPU, general performance, etc. Then maybe you decide to cache, but then the user may find it weird that some minor app takes up as much size as their photos.

The effect of all this is that you need to plan and build this into your expectations - don't promise results the day after a release if you're evaluating a client-side change. Adjust for the fact that your results are conditional and gather data on those conditions e.g. battery level or whatever. Adjust in your analysis for different devices and platforms, etc. This is a lovely opportunity for multilevel regression.

### 9.8.3 Instrumentation

Drawing on [Kohavi et al. \(2020\)](#), p. 162 - 165), I'll now discuss instrumentation. [Kohavi et al. \(2020\)](#) use the name 'instrumentation'. I'd prefer something like 'measurement methods' so that we don't confuse this with the entirely different concept of instrumental variables later in the course, but instrumentation is what is used in industry, so we'll use that here too.

Regardless of what it's called, the point of this is that you need to consider how you are getting your data in the first place. For instance, if we put a cookie on your device then different types of users will remove that at different rates. Using things like beacons can be great (this is when you force the user to 'download' some tiny thing they don't notice so that you know they've gone somewhere - see 'email' etc). But again, there are practical issues - do we force the beacon before the main content loads - which makes for a worse customer experience; or do we allow the beacon to load after the main content, in which case we may get a biased sample?

There are likely different servers and databases for different faces of the product. For instance, Twitter in Australia, compared with Twitter in Canada, compared with Twitter on my phone's app, compared with Twitter accessed via the browser. Joining these different datasets can be difficult and requires either a unique id or some probabilistic approach.

[Kohavi et al. \(2020\)](#), p. 165) recommend changing the culture of your workplace to ensure instrumentation is normalised, which I mean, yeah, good luck.

### 9.8.4 Randomisation unit

Again, drawing on [Kohavi et al. \(2020\)](#), p. 162 - 165), we need to be very aware of what are we actually randomising over? Again, this is something that's kind of obvious in normal RCTs, but gets like really interesting in the case of A/B testing. Let's consider the malaria netting experiments - either a person/village/state gets a net or it doesn't. Easy (relatively). But in the

case of server-side A/B testing - are we randomising the page, the session, or the user?

To think about this, let's think about colour. Let's say that we change our logo from red to blue on the 'home' page. If we're randomising at the page level, then when the user goes to the 'about' page the logo could be back to red. If we're randomising at the session level, then it'll be blue while they're using the website that time, but if they close it and come back then it'll be red. Finally, if we're randomising at a user level then it'll always be red for me, but always blue for my friend. That last bit assumes perfect identity tracking, which might be generally okay if you're Google or Facebook, but for anyone else is going to be a challenge - what if you visit cbc.ca on your phone and then on your laptop? You're likely considered a different 'user'.

Does this matter? It's a trade-off between consistency and importance.

We are always interested in whether the treatment and control groups have been created randomly. One way to test it is an A/A test. [Taddy \(2019, p. 129\)](#) describes how 'AB platforms typically run "AA" tests that show the same website in groups A and B. If you see a significant difference between groups in an AA trial, then something is likely wrong in your randomization.'

[Kohavi et al. \(2020, p.201\)](#) says similarly, that '(w)e highly recommend running continuous A/A tests in parallel with other experiments to uncover problems, including distribution mismatches and platform anomalies.'

### 9.8.5 Partnerships

Unless we work at a Facebook/Twitter type firm, it may not be possible to run A/B tests ourselves at scale. While we can randomise our own personal website fairly easily, for most of us there won't be many visitors. Hence it can be important to partner with such firms. [Salganik \(2018, p. 187\)](#) draws our attention to the fact that there may be tension between 'the researchers and the partners'.

As an example, [Salganik \(2018\)](#) discusses a situation where one treatment (out of the three that were possible) accounted for 98 per cent of the sample because Facebook wanted to treat everyone. The researchers were only able to convince 'them to hold back 1 per cent for a related treatment and 1 per cent for a control group.' He continues:

---

without the control group, it would have been basically impossible to measure the effect of the Info + Social treatment because it would have been a "perturb and observe" experiment, rather than a randomized controlled experiment. This example

provides a valuable practical lesson for working with partners: sometimes you create an experiment by convincing someone to deliver a treatment and sometimes you create an experiment by convincing someone not to deliver a treatment (i.e. to create a control group).

Salganik (2018, p. 188).

---

In order to identify such opportunities, Salganik (2018, p. 188) advises us ‘to notice a real problem that you can solve while you are doing interesting science’. Salganik (2018) closes with four other pieces of advice:

- 1) ‘(Y)ou should think as much as possible before any data have been collected’ Salganik (2018, p. 189).
- 2) ‘(Y)ou should consider designing a series of experiments that reinforce each other’ Salganik (2018, p. 190).
- 3) You should ‘(c)reate zero variable cost data’, by: 1) trying to replace human work with computer work; and 2) creating fun experiments that participants want to participate in (Salganik, 2018, p. 191).
- 4) You should ‘(b)uild ethics into your design: replace, refine, and reduce’, that is ‘(m)ake your experiment more humane by replacing experiments with non-experimental studies, refining the treatments [to be as harmless as possible], and reducing the number of participants’ (Salganik, 2018, p. 196).

### 9.8.6 Speed vs quality

Don’t peek at your results early and then call off the rest of the experiment if you’ve got significance. You essentially ruin everything that underpins statistics if you do that.

### 9.8.7 Conflicting priorities

One of the interesting aspects of A/B testing is that we’re usually running them not because we desperately care about the specific outcome, but because that feeds into some other measure that we care about. For instance, do we care whether the website is quite-dark-blue or slightly-darker-blue or white? Probably not, but we probably care a lot about the company share price. But then what if picking the best blue comes at a cost to the share price?

Obviously, this is a bit contrived, so let’s pretend that we work at a food delivery app and that we’re the junior data scientist in charge of driver satisfaction. We do some A/B tests and we find that drivers are always happier

when they are able to deliver food to the customer faster. Faster is better, always. But one way to achieve faster deliveries, is for them to not put the food into a hot box that will maintain the temperature. Something like that might save 30 seconds, which is significant on a 10-15 minute deliver. Unfortunately, although making a decision like that on the basis of A/B tests designed to optimize driver-satisfaction, would ultimately likely make the customer experience worse. If customers receive cold food, (when it's meant to be hot) then they may stop using the service and so this is likely bad for the app in the longer term.

This trade-off may be obvious if you're running the driver-experiment and you're looking at the customer complaints. Maybe on a small team or in a start-up you would be. But if you work for a larger team, you'd likely not and so ensuring that A/B tests aren't resulting in false optimization is something that is especially interesting, and not a typical trade-off in a normal RCT.

## 9.9 Case study - Upworthy

The trouble with much of A/B testing is that because it's done by firms we typically don't have datasets that we can use. However, J. Nathan Matias (Cornell), Kevin Munger (Penn State), and Marianne Aubin Le Quere (Cornell) obtained a dataset of A/B tests from Upworthy that they provide access to (Matias et al., 2019). You are able to request access to the dataset here: <https://upworthy.natematias.com> (this request may take a couple of weeks to be processed). Upworthy was a click-bait news company that used A/B testing to optimize their content. More details are provided by Fitts (2014).

Let's have a quick look at the data.

```
upworthy <- read_csv(here::here("dont_push/upworthy-archive-exploratory-packages-03.12.2020.csv"))

upworthy %>%
 head()

#> # A tibble: 6 x 17
#> ...1 created_at updated_at clickability_test_id excerpt
#> <dbl> <dttm> <dttm> <chr> <chr>
#> 1 0 2014-11-20 06:43:16 2016-04-02 16:33:38 546d88fb84ad38b2ce000024 Things~
#> 2 1 2014-11-20 06:43:44 2016-04-02 16:25:54 546d88fb84ad38b2ce000024 Things~
#> 3 2 2014-11-20 06:44:59 2016-04-02 16:25:54 546d88fb84ad38b2ce000024 Things~
#> 4 3 2014-11-20 06:54:36 2016-04-02 16:25:54 546d902c26714c6c44000039 Things~
#> 5 4 2014-11-20 06:54:57 2016-04-02 16:31:45 546d902c26714c6c44000039 Things~
#> 6 5 2014-11-20 06:55:07 2016-04-02 16:25:54 546d902c26714c6c44000039 Things~
```

```
#> # ... with 12 more variables: headline <chr>, lede <chr>, slug <chr>,
#> # eyecatcher_id <chr>, impressions <dbl>, clicks <dbl>, significance <dbl>,
#> # first_place <lgl>, winner <lgl>, share_text <chr>, square <chr>,
#> # test_week <dbl>

upworthy %>%
 names()
#> [1] "...1" "created_at" "updated_at"
#> [4] "clickability_test_id" "excerpt" "headline"
#> [7] "lede" "slug" "eyecatcher_id"
#> [10] "impressions" "clicks" "significance"
#> [13] "first_place" "winner" "share_text"
#> [16] "square" "test_week"
```

From the documentation: ‘The Upworthy Research Archive contains packages within tests. On Upworthy, packages are bundles of headlines and images that were randomly assigned to people on the website as part of a test. Tests can include many packages.’ So each row is a package and it should be part of a test ‘clickability\_test\_id’.

We have a variety of variables. We’ll focus on ‘created\_at’, ‘clickability\_test\_id’ so that we can create comparison groups, ‘headline’, ‘impressions’ which is the number of people that saw the package, and ‘clicks’ which is the number that clicked on that package. So within each batch of tests, we’re interested in the effect of varied headlines on impressions and clicks.

```
upworthy_restricted <-
 upworthy %>%
 select(created_at, clickability_test_id, headline, impressions, clicks)

head(upworthy_restricted)
#> # A tibble: 6 x 5
#> created_at clickability_test_id headline impressions clicks
#> <dttm> <chr> <chr> <dbl> <dbl>
#> 1 2014-11-20 06:43:16 546d88fb84ad38b2ce000024 They're Being~ 3052 150
#> 2 2014-11-20 06:43:44 546d88fb84ad38b2ce000024 They're Being~ 3033 122
#> 3 2014-11-20 06:44:59 546d88fb84ad38b2ce000024 They're Being~ 3092 110
#> 4 2014-11-20 06:54:36 546d902c26714c6c44000039 This Is What ~ 3526 90
#> 5 2014-11-20 06:54:57 546d902c26714c6c44000039 This Is What ~ 3506 120
#> 6 2014-11-20 06:55:07 546d902c26714c6c44000039 This Is What ~ 3380 98
```

We are going to focus on the text contained in headlines. We also want to remove the effect of different pictures, by comparing on the same image. I’m

interested in whether headlines that asked a question got more clicks than those that didn't.

To identify whether a headline asks a question, I'm going to just search for a question mark. Although there are more complicated constructions that we could use, this will be enough to get started.

```
upworthy_restricted <-
 upworthy_restricted %>%
 mutate(asks_question = stringr::str_detect(string = headline, pattern = "\\\?"))

upworthy_restricted %>% count(asks_question)
#> # A tibble: 2 x 2
#> asks_question n
#> <lgl> <int>
#> 1 FALSE 19130
#> 2 TRUE 3536
```

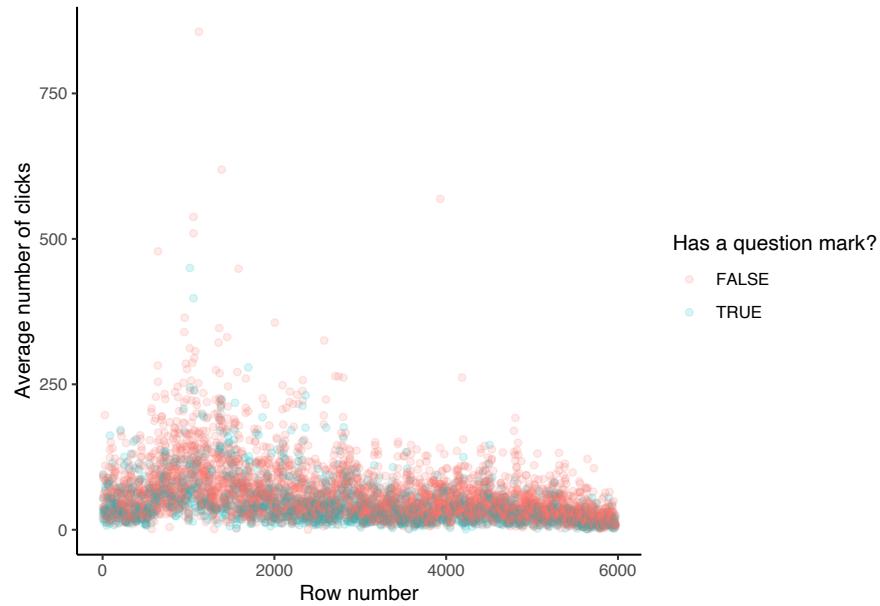
Now for every test, for every picture, we want to know whether asking a question affected the number of clicks.

```
to_question_or_not_to_question <-
 upworthy_restricted %>%
 group_by(clickability_test_id, asks_question) %>%
 summarise(ave_clicks = mean(clicks)) %>%
 ungroup()

look_at_differences <-
 to_question_or_not_to_question %>%
 pivot_wider(id_cols = clickability_test_id,
 names_from = asks_question,
 values_from = ave_clicks) %>%
 rename(ave_clicks_not_question = `FALSE`,
 ave_clicks_is_question = `TRUE`) %>%
 filter(!is.na(ave_clicks_not_question)) %>%
 filter(!is.na(ave_clicks_is_question)) %>%
 mutate(difference_in_clicks = ave_clicks_is_question - ave_clicks_not_question)

look_at_differences$difference_in_clicks %>% mean()
#> [1] -4.890435
```

So we find that in general, having a question in the headline may slightly decrease the number of clicks on a headline, although if there is an effect it does not appear to be very large (Figure 9.6).



**FIGURE 9.6:** Comparison of the average number of clicks when a headline contains a question mark or not.

## 9.10 Implementing surveys

### 9.10.1 Google

### 9.10.2 Facebook

### 9.10.3 Survey Monkey

### 9.10.4 Mechanical Turk

### 9.10.5 Prolific

### 9.10.6 Qualtrics

### 9.10.7 Other

---

## 9.11 Sensor data

---

## 9.12 FOI

- 2019. Walby, Kevin and Alex Luscombe, eds. Freedom of Information and Social Science Research Design. New York: Routledge.
- 

## 9.13 Next steps

Large scale experiments are happening all around us. These days I feel we all know a lot more about healthcare experiments than perhaps we'd like to know and the AstraZeneca/Oxford situation is especially interesting, for instance, [Oxford-AstraZeneca \(2020\)](#), but see [Bastian \(2020\)](#) for how this is actually possibly more complicated.

There are also well-known experiments that tried to see if big government programs are effective, such as:

- The RAND Health Insurance Experiment randomly gave health insurance to people in the US between 1974 and 1982 ([Brook et al., 1984](#)).
- The Oregon Health Study randomly gave health insurance in Oregon in 2008 ([Finkelstein et al., 2012](#)).

## 9.14 Exercises and tutorial

### 9.14.1 Exercises

1. In your own words, what is the role of randomisation in constructing a counterfactual (write two or three paragraphs)?
2. What is external validity (pick one)?
  - a. Findings from an experiment hold in that setting.
  - b. Findings from an experiment hold outside that setting.
  - c. Findings from an experiment that has been repeated many times.
  - d. Findings from an experiment for which code and data are available.
3. What is internal validity (pick one)?
  - a. Findings from an experiment hold in that setting.
  - b. Findings from an experiment hold outside that setting.
  - c. Findings from an experiment that has been repeated many times.
  - d. Findings from an experiment for which code and data are available.
4. If we have a dataset named ‘netflix\_data’, with the columns ‘person’ and ‘tv\_show’ and ‘hours’, (person is a character class uniqueID for every person, tv\_show is a character class name of a tv show, and hours is double expressing the number of hours that person watched that tv show). Could you please write some code that would randomly assign people into one of two groups? The data looks like this:

```
library(tidyverse)
netflix_data <-
 tibble(person = c("Rohan", "Rohan", "Monica", "Monica", "Monica",
 "Patricia", "Patricia", "Helen"),
 tv_show = c("Broadchurch", "Duty-Shame", "Broadchurch", "Duty-Shame",
 "Shetland", "Broadchurch", "Shetland", "Duty-Shame"),
 hours = c(6.8, 8.0, 0.8, 9.2, 3.2, 4.0, 0.2, 10.2)
)
```

5. In the context of randomisation, what does stratification mean to you (write a paragraph or two)?
6. How could you check that your randomisation had been done appropriately (write two or three paragraphs)?
7. Identify three companies that conduct A/B testing commercially

and write a short paper about how they work and the trade-offs of each. Are there any notable Toronto-based or Canadian companies? Why do you think this might be the case?

8. Pretend that you work as a junior analyst for a large consulting firm. Further, pretend that your consulting firm has taken a contract to put together a facial recognition model for the Canada Border Services Agency's Inland Enforcement branch. Taking a page or two, please discuss your thoughts on this matter. What would you do and why?
9. What are some types of probability sampling, and in what circumstances might you want to implement them (write two or three pages)?
10. There have been some substantial political polling 'misses' in recent years (Trump and Brexit come to mind). To what extent do you think non-response bias was the cause of this (write a page or two, being sure to ground your writing with citations)?
11. What is an estimate (pick one)?
  - a. A rule for calculating an estimate of a given quantity based on observed data.
  - b. The quantity of interest.
  - c. The result.
  - d. Unknown numbers that determine a statistical model.
12. What is an estimator (pick one)?
  - a. A rule for calculating an estimate of a given quantity based on observed data.
  - b. The quantity of interest.
  - c. The result.
  - d. Unknown numbers that determine a statistical model.
13. What is an estimand (pick one)?
  - a. A rule for calculating an estimate of a given quantity based on observed data.
  - b. The quantity of interest.
  - c. The result.
  - d. Unknown numbers that determine a statistical model.
14. What is a parameter (pick one)?
  - a. A rule for calculating an estimate of a given quantity based on observed data.
  - b. The quantity of interest.
  - c. The result.
  - d. Unknown numbers that determine a statistical model.
15. It seems like a lot of businesses have closed in downtown Toronto since the pandemic. To investigate this, I decide to walk along some blocks downtown and count the number of businesses that are closed and open. To decide which blocks to walk, I open a map of Toronto,

- start at the lake, and then pick every 10th street. This type of sampling is (select all)?
- a. Cluster sampling.
  - b. Systematic sampling.
  - c. Stratified sampling.
  - d. Simple random sampling.
  - e. Convenience sampling.
16. Please name some reasons why you may wish to use cluster sampling (select all)?
- a. Balance in responses.
  - b. Administrative convenience.
  - c. Efficiency in terms of money.
  - d. Underlying systematic concerns.
  - e. Estimation of sub-populations.
17. Please consider Beaumont, 2020, ‘Are probability surveys bound to disappear for the production of official statistics?’. With reference to that paper, do you think that probability surveys will disappear, and why or why not (please write a paragraph or two)?
18. [Ware \(1989\)](#), p. 298) mentions ‘a randomized play the winner design’. What is it?
19. [Ware \(1989\)](#), p. 299) mentions ‘adaptive randomization’. What is it, in your own words?
20. [Ware \(1989\)](#), p. 299) mentions ‘randomized-consent’. He continues that it was ‘attractive in this setting because a standard approach to informed consent would require that parents of infants near death be approached to give informed consent for an invasive surgical procedure that would then, in some instances, not be administered. Those familiar with the agonizing experience of having a child in a neonatal intensive care unit can appreciate that the process of obtaining informed consent would be both frightening and stressful to parents’. To what extent do you agree with this position, especially given, as Ware (1989), p. 305, mentions ‘the need to withhold information about the study from parents of infants receiving CMT’?
21. [Ware \(1989\)](#), p. 300) mentions ‘equipoise’. In your own words, please define and discuss it, using an example from your own experience.
22. What is power (in a statistical context)?

### 9.14.2 Tutorial

The purpose of this tutorial is to ensure that it is clear in your mind how thoroughly you should know your dataset. It builds on the ‘memory palace’ technique used by professional memorisers, as described by [Foer \(2011\)](#).

Please think about your childhood home, or another building that you know intimately. Imagine yourself standing at the front of it. Describe what it looks

like. Then ‘walk’ into the front and throughout the house, again describing each aspect in as much detail as you can imagine. What are each of the rooms used for and what are their distinguishing features? How does it smell? What does this all evoke in you? Please write a page or two.

Now think about a dataset that you’re interested in. Please do this same exercise, but for the dataset.



# 10

---

## *Farm data*

---

**STATUS:** Under construction.

### Required reading

- [Crawford \(2021\)](#), Chapter 3.
- [Statistics Canada \(2017\)](#), Chapter 10 - Data quality assessment.

### Recommended reading

- 

### Key concepts/skills/etc

- 

### Key libraries

- 

### Key functions/etc

- 

---

### 10.1 Overview

There are a variety of sources of data that have been produced for the purposes of being used as datasets. One thinks here especially of censuses. [Whitby \(2020, p. 30-31\)](#) provides an enthralling overview, describing how '(t)he earliest censuses suggested in writing come... from China's Yellow River valley' and that they were used for more than just purposes of taxation and conscription. He continues, highlighting the links between the census and Christianity, for instance from the Book of Luke 'In those days Caesar Augustus issued a decree that a census should be taken of the entire Roman world', which led to David and Mary travelling to Bethlehem.

Taxation was a substantial motivator for censuses. [Jones \(1953\)](#) describes how census records survive that 'were probably engraved in the late third or early fourth century A.D., when Diocletian and his colleagues and successors are

known to have been active in carrying out censuses to serve as the basis of their new system of taxation'. And detailed records of this sort have been abused. For instance, Luebke and Milton (1994) say how '(t)he Nazi regime gathered its information with two relatively conventional tools of modern administration: the national census and police registration'.

Another source of data deliberately put together to be a dataset include economic conditions such as unemployment, inflation, and GDP. Interestingly, Rockoff (2019) describes how these economic statistics were not actually developed by the federal government, even though it 'eventually took over the role of producing these statistics on a regular basis.' The broader point is that these types of datasets and censuses are typically put together by governments. They have the powers of the state behind them.

Another, similarly large and established source of data are from long-running large surveys. These are conducted on a regular basis, and while not usually directly conducted by the government, they are usually funded, one way or another, by the government. For instance, here we often think of electoral surveys, such as the Canadian Election Study, which has run in association with every federal election since 1965, and similarly the British Election Study which has been associated with every general election since 1964.

Finally, there has been a large push toward open data in government. While the term has become contentious because of how it has occurred in practice, the underlying principle - that the government should make available the data that it has - is undeniable.

In this chapter we cover these datasets, which I term 'farmed data'. They are typically fairly nicely put together and the work of collecting, preparing and cleaning these datasets has typically been done. They are also, usually, conducted on a set release cycle. For instance, most developed countries release unemployment and inflation dataset on a monthly basis, GDP on a quarterly basis, and a census every five to ten years.

While these datasets have always been useful, they were developed for a time when much analysis was conducted without the use of scripts and programming languages. A cottage industry of R package development has sprung up around making it easier to get these datasets into R. In this chapter we cover a few that are especially useful.

It is important to recognise that data are not neutral. Thinking clearly about who is included in the dataset, and who is systematically excluded, is critical. As Crawford (2021, p. 121) says:

---

The way data is understood, captured, classified, and named is

fundamentally an act of world-making and containment. It has enormous ramifications for the way artificial intelligence works in the world and which communities are most affected. The myth of data collection as a benevolent practice in computer science has obscured its operations of power, protecting those who profit most while avoiding responsibility for its consequences.

---

---

## 10.2 Censuses

MEasuring homelessness?

<https://www.ncbi.nlm.nih.gov/books/NBK218229/>

“S-Night survey conducted by the US Census Bureau in 1990”

“Street count surveys are used in many cities to count the number of homeless people in the streets at a point in time and gain a better understanding of the needs of homeless populations. In surveys such as the S-Night survey conducted by the US Census Bureau in 1990, enumerators are sent to pre-identified sites to enumerate homeless people while other survey personnel (“plants”) are planted among homeless people and indistinguishable from actual homeless. The ratio of plants seen by the enumerators to the number of plants deployed is used to inform the detection probability of homeless and provide an adjustment to the homeless undercount. In practice, one cannot know for sure which plants were seen because enumerators cannot distinguish between plants and homeless people. We can only rely on the plants’ judgement of whether (yes, no or maybe) they think they were seen by an enumerator. The presence of “maybes” in the data leads to more unknown parameters than data points, which makes estimation of detection probabilities difficult. We propose to solve this problem by developing a Bayesian hierarchical model that uses hierarchical priors on detection probabilities across survey years and/or across cities. Such hierarchical modeling of the data is challenging because the data is available at various aggregated levels of the population (e.g., *seen* = plant AND *seen* + homeless AND *seen*, *maybes* = plant AND *seen* AND not interviewed AND maybe + plant AND not *seen* AND maybe.) The new methodology will be applied to a simulated reconstruction of the original S-night survey data and our estimates will be compared to those of the original analysis of the S-night data.”

### 10.2.1 Canada

The first census in Canada was conducted in 1666. There were 3,215 inhabitants that were counted and the census ‘recorded their age, sex, marital status and occupation’ (Statistics Canada, 2017). In 1867 a decennial census was required to ‘determine representation by population in the new Parliament.’ Regular censuses have occurred since then, the most recent in 2021.

In general, data from the Canadian census is not as easily available as in other countries. The ‘Individuals File, 2016 Census Public Use Microdata Files (PUMF)’ - <https://www150.statcan.gc.ca/n1/en/catalogue/98M0001X> - is probably the best first step, although it does not provide much detail. It is a 2.7 per cent sample from the 2016 census. It is free, but must be requested, after which Statistics Canada will email access details.

Another way to access data from the Canadian census is to use `cancensus`, which is an R package that provides access to the Canadian census (von Bergmann et al., 2021). The use of this package requires an API key, which can be requested by creating an account here: [https://censusmapper.ca/users/sign\\_up](https://censusmapper.ca/users/sign_up) and then clicking ‘edit profile’. The package has a helper function `cancensus::set_api_key("ADD_YOUR_API_KEY_HERE", install = TRUE)` that makes it easier to add the key to your ‘Renvironment’ file. Once that is done you can use the package to access the data.

The main function in the package is `cancensus::get_census()`, and that requires an argument for the census of interest, and a variety of other factors. In this example we will use the 2016 census

```
library(tidyverse)
library(cancensus)

ontario_population <-
 get_census(dataset = "CA16",
 level = "Regions",
 vectors = "v_CA16_1",
 regions = list(PR=c('35'))
)
#>
Downloading: 170 B
```

```
head(ontario_population)
#> # A tibble: 1 × 9
#> GeoUID Type `Region Name` `Area (sq km)` Population Dwellings Households
#> <chr> <fct> <fct> <dbl> <dbl> <dbl> <dbl>
#> 1 35 PR Ontario 986722. 13448494 5598391 5169174
#> # ... with 2 more variables: C_UID <chr>, v_CA16_1: Age Stats <dbl>
```

The package is fiddly, however it is worthwhile to ensure a reproducible workflow if the data that you are interested in are included. The two helper functions—`list_census_regions()` and `list_census_vectors()` — may be useful identify the arguments of interest.

### 10.2.2 USA

The requirement for a US Census is included in their constitution, and there is fairly decent, although chunky, general access. However, the US is in an interesting situation where there may actually be better options. For instance, the American Community Survey (ACS) is a survey that is comparable to the questions asked on many censuses that is conducted monthly. By the end of the year, it ends up with millions of responses. Although the ACS is smaller than a census, the advantage is that it is available on a more timely basis.

The best way to access the ACS is to use the Integrated Public Use Microdata Series (IPUMS). You will need to create an account, but it is free and easy to do.

Lol, just use IPUMS.

### 10.2.3 Australia

---

## 10.3 Other government data

### 10.3.1 Unemployment

### 10.3.2 Weather

### 10.3.3 That Canadian survey that we used in STA304 from the library

---

## 10.4 Open Government Data

### 10.4.1 Canada

- Canadian Government Open Data: <https://open.canada.ca/en/open-data>.
  - City of Toronto Open Data Portal
- 

## 10.5 Electoral Studies

### 10.5.1 Canadian Electoral Study

### 10.5.2 Australian Electoral Study

### 10.5.3 CCES

---

## 10.6 Other

- University of Toronto Dataverse: <https://dataverse.scholarsportal.info/dataverse/toronto>.
- Data is Plural structured archive: <https://docs.google.com/spreadsheets/d/1wZhPLMCHKJvwOkP4juclhjFgqIY8fQFMemwKL2c64vk/edit#gid=0>.
- Kaggle Datasets: <https://www.kaggle.com/datasets>.
- Figure Eight: <https://www.figure-eight.com/data-for-everyone/>.
- Google dataset search: <https://datasetsearch.research.google.com/>.
- Awesome Data: <https://github.com/awesomedata/awesome-public-datasets>.

## 10.7 Exercises and tutorial

### 10.7.1 Exercises

1. Please identify three other sources of data that you are interested in and describe where they are available (please include a link or code)?
2. Please focus on one of those sources. What steps do you have to go through in order to get a dataset that can be analysed in R?
3. Let's say you take a job at RBC (a Canadian bank) and they already have some quantitative data for you to use. What are some questions that you should explore when deciding whether that data will be useful to you?
4. Write three points (you are welcome to use dot points) about why government data may be especially useful?
5. Please pick a government of interest and find their inflation statistics. To what extent do you know about how these data were gathered?
6. With reference to Chen et al. (2019) and Martinez (2019) to what extent do you think we can trust government statistics? Please mention at least three governments in your answer.
7. The 2021 census in Canada asked, firstly, 'What was this person's sex at birth? Sex refers to sex assigned at birth. Male/Female', and then 'What is this person's gender? Refers to current gender which may be different from sex assigned at birth and may be different from what is indicated on legal documents. Male/Female/Or please specify this person's gender (space for a typed or handwritten answer)'. With reference to Statistics Canada (2020), please discuss the extent to which you think this is an appropriate way for census to have proceeded. You are welcome to discuss the case of a different country if you are more familiar with that.
8. Pretend that we have conducted a survey of everyone in Canada, where we asked for age, sex, and gender. Your friend claims that there is no need to worry about uncertainty 'because we have the whole population'. Is your friend right or wrong, and why?

### 10.7.2 Tutorial



---

---

## Part IV

# Preparation



# 11

---

## *Cleaning and preparing data*

---

**STATUS:** Under construction.

### Required reading

- *Data Feminism*, Chapter 5 ([D'Ignazio and Klein, 2020](#)).
- *R for Data Science*, Chapter 9 ([Wickham and Grolemund, 2017](#)).

### Recommended reading

- *We Gave Four Good Pollsters the Same Raw Data. They Had Four Different Results*, ([Cohn, 2016](#)).
- *Data Cleaning Procedures for the 1993 Robert Wood Johnson Foundation Employer Health Insurance Survey*, ([Euller et al., 1997](#)).
- *Data Cleaning*, ([Ilyas and Chu, 2019](#)).

### Key concepts/skills/etc

- Planning an end-point, and simulating the data that you'd like, are key elements of cleaning and preparing data.
- Begin on a small sample of the dataset, write code to fix that, and then iterate and generalize to additional tranches.
- Develop a series of tests and checks that your final dataset should pass so that the features of the dataset are clear.
- Be especially concerned about the class of variables, having clear names, and that the values of each variable are as expected given all this.

### Key libraries

- `janitor` ([Firke, 2020](#))
- `stringr` ([Wickham, 2019e](#))
- `tidyverse` ([Wickham, 2021b](#))

### Key functions/etc

- `dplyr::count()`
- `dplyr::mutate()`
- `dplyr::select()`
- `janitor::clean_names()`
- `stringr::str_replace_all()`
- `stringr::str_trim()`

- `tidy::pivot_longer()`
  - `tidy::separate()`
  - `tidy::separate_rows()`
- 

## 11.1 Introduction

---

“Well, Lyndon, you may be right and they may be every bit as intelligent as you say,” said Rayburn, “but I’d feel a whole lot better about them if just one of them had run for sheriff once.”

Sam Rayburn’s reaction to Lyndon Johnson’s enthusiasm about Kennedy’s incoming cabinet, as quoted by [Halberstam \(1972, p. 41\)](#).

---

In earlier chapters we have done data cleaning and preparation, but in this chapter, we get into the weeds. I do not trust anyone who works with data who has not spent time, at some point in their career, cleaning data. And by the end of this chapter I think that you’ll feel the same way. To clean and prepare data is to make a lot of different decisions, many of which are important.

For a long time, data cleaning and preparation was largely overlooked. We now realise that was a mistake. It is no longer possible to trust almost any result in disciplines that apply statistics. The reproducibility crisis, which started in psychology but has now extended to many other fields in the physical and social sciences, has brought to light issues such as p-value ‘hacking’, researcher degrees of freedom, file-drawer issues, and even data and results fabrication ([Gelman and Loken, 2013](#)). Steps are now being put in place to address these. However, there has been relatively little focus on the data gathering, cleaning, and preparation aspects of applied statistics, despite evidence that decisions made during these steps greatly affect statistical results ([Huntington-Klein et al., 2020](#)). In this chapter we focus on these issues.

While the statistical practices that underpin data science are themselves correct and robust when applied to ‘fake’ datasets created in a simulated environment, data science is typically not conducted with these types of datasets. For instance, data scientists are interested in:

---

...the kind of messy, unfiltered, and possibly unclean data—tainted by heteroskedasticity, complex dependence and missingness patterns—that until recently were avoided in polite conversations between more traditional statisticians...

(Craiu, 2019).

---

Big data does not resolve this issue, and may even exacerbate it, for instance ‘without taking data quality into account, population inferences with Big Data are subject to a Big Data Paradox: the more the data, the surer we fool ourselves’ (Meng, 2018). It is important to note that the issues that are found in much applied statistics research are not necessarily associated with researcher quality, or their biases (Silberzahn et al., 2018). Instead, they are a result of the environment within which data science is conducted. This chapter aims to give you the tools to explicitly think about this work.

Gelman and Vehtari (2020) writing about the most important statistical ideas of the past 50 years say that:

---

...each of them was not so much a method for solving an existing problem, as an opening to new ways of thinking about statistics and new ways of data analysis. To put it another way, each of these ideas was a codification, bringing inside the tent an approach that had been considered more a matter of taste or philosophy than statistics.

---

I see the focus on data cleaning and preparation in this chapter as analogous, insofar, as it represents a codification, or bringing inside the tent, of aspects that are typically (incorrectly) considered those of taste rather than statistics.

The approach that I recommend that you follow is:

1. Duplicate.
2. Plan the end state.
3. Execute that plan on a tiny sample.
4. Write tests and documentation.
5. Iterate the plan.
6. Generalize the execution.
7. Update tests and documentation.

You will need to use all your skills to this point to be effective, but this is the very stuff of statistical sciences! Be dogged, but sensible. The best is the enemy of the good here. It's better to have 90 per cent of the data cleaned and prepared, and to start exploring that, before deciding whether it's worth the effort to clean and prepare the remaining 10 per cent because that remainder will likely take an awful lot of time and effort.

As [Van den Broeck et al. \(2005\)](#) say, all data regardless of whether they were obtained from hunting, gathering, or farming, will have issues and it is critical that you understand how to 'deal with errors from various sources' and understand 'their effects on study results'. To clean data is to analyze data. As [Au \(2020\)](#) says 'The act of cleaning data is the act of preferentially transforming data so that your chosen analysis algorithm produces interpretable results. That is also the act of data analysis.' We are attempting to triangulate the situation.

---

## 11.2 Workflow

### 11.2.1 Save a copy of the raw data

The first step is to save the raw data into a separate folder location. It is critical that you save the raw data to the extent that is possible ([Wilson et al., 2017](#)). In an ideal situation, duplicate the folder that contains the raw data, and then rename the duplicated folder 'cleaned' and then only ever modify the data in that cleaned folder.

### 11.2.2 Begin with an end in mind

Planning the end state, or forcing yourself to begin with an end in mind is important for a variety of reasons. As with scraping data, it helps us to be pre-active about scope-creep, but with data cleaning I see a bigger benefit being that it forces us to really think about what we want the final dataset to look like. As before, I recommend first sketching the dataset. The key features of your sketch will be aspects such as the names of the columns, their class, and the possible range of values. It might look something like Figure 11.1 .

Notice that this process has made it clear that we want the full names of the states, rather than abbreviations. And that population should be in millions, rather than some other units. The process of sketching out an end-point forces us to make decisions and be clear about our desired end state.

We then implement that using code to simulate data. Again, this process forces us to think about what reasonable values look like in our dataset because we are literally forced to decide which functions to use. Thinking carefully about

State	Population
Alabama	5
Alaska	0.7
:	:
:	:

**FIGURE 11.1:** Example of a dataset end plan

the membership of each column here, for instance if the column is meant to be ‘gender’ then values such as ‘male’, ‘female’, ‘other’, and ‘unknown’ may be expected, but a number such as ‘1,000’ would likely be unexpected. It also forces us to be explicit about variable names because we have to assign the outputs of those functions to a variable.

Building on the example above, perhaps it might look something like this.

```
library(tidyverse)

simulated_states_population <-
 tibble(
 state = c('Alabama', 'Alaska', 'Arizona', 'Arkansas', 'California'),
 population = c(5, 0.7, 7, 3, 40)
)

simulated_states_population
#> # A tibble: 5 x 2
#> state population
#> <chr> <dbl>
#> 1 Alabama 5
#> 2 Alaska 0.7
```

```
#> 3 Arizona 7
#> 4 Arkansas 3
#> 5 California 40
```

Our purpose, during data cleaning and preparation is to then bring our dataset close to that plan.

The desired end-state will typically be ‘tidy data’ where (Wickham, 2014, p. 4):

1. Each variable is a column
2. Each observation is a row
3. Each observational unit is a table.

The following is an example of data in a tidy format:

```
tibble(
 name = c("Alfred", "Ben", "Chris", "Daniela", "Emma"),
 age_group = c("0-9", "10-19", "0-9", "0-9", "10-19"),
 response_time = c(0, 9, 3, 3, 8),
)
#> # A tibble: 5 x 3
#> name age_group response_time
#> <chr> <chr> <dbl>
#> 1 Alfred 0-9 0
#> 2 Ben 10-19 9
#> 3 Chris 0-9 3
#> 4 Daniela 0-9 3
#> 5 Emma 10-19 8
```

### 11.2.3 Start small

At this point, we then have our first look at the data that we are dealing with. Regardless of what the raw data look like we want to try to get it into a rectangular dataset as quickly as possible because then we can use our familiar verbs. Let’s assume here that we’re starting with some .txt file.

The first step is to look for regularities in the dataset. We are wanting to end up with tabular data, which means that we need some type of delimiter to distinguish different columns. Ideally this might be features such as a comma, a semicolon, a tab, a double space, or a line break.

```
Alabama, 5 million.
Alaska, 0.7 million.
```

```
Arizona, 7 million.
Arkansas, 3 million.
California, 40 million.
```

In worse cases there may be a feature of the dataset that we can take advantage of. For instance, possibly the data look like the following:

```
State is Alabama and population is 5 million.
State is Alaska and population is 0.7 million.
State is Arizona and population is 7 million.
State is Arkansas and population is 3 million.
State is California and population is 40 million.
```

In this case, although we don't have a traditional delimiter we can use the regularity of 'State is' and ' and population is ' to get what we need.

A more difficult case is this following:

```
Alabama 5 Alaska 0.7 Arizona 7 Arkansas 3 California 40
```

One way to approach this is to take advantage of the different classes and values that we're looking for. For instance, in this case, we know that we're after US states, so there are only 50 possible options and we could use the existence of these as a delimiter. We could also use the fact that population is a number here, and so split based on a space followed by a number.

To move forward, I'll assume that last example and move it into a rectangular dataset.

```
raw_data <- c('Alabama 5 Alaska 0.7 Arizona 7 Arkansas 3 California 40')

data <-
 tibble(raw = raw_data)

data <-
 data %>%
 tidyrr::separate(col = raw,
 into = letters[1:5],
 sep = "(?=<=[[:digit:]]) ") %>%
 pivot_longer(cols = letters[1:5],
 names_to = "drop_me",
 values_to = "separate_me") %>%
 tidyrr::separate(col = separate_me,
 into = c('state', 'population'),
```

```

sep = " (?=[[:digit:]])" %>%
select(-drop_me)

data
#> # A tibble: 5 x 2
#> state population
#> <chr> <chr>
#> 1 Alabama 5
#> 2 Alaska 0.7
#> 3 Arizona 7
#> 4 Arkansas 3
#> 5 California 40

```

#### 11.2.4 Write tests and documentation.

Having established a rectangular dataset, albeit a messy one, we should begin to look at the classes that we have. We don't necessarily want to fix the classes at this point, because that can result in us losing data. But we look at the class to see what it is, and then compare it to our simulated dataset to see where it needs to get to. We note the columns where it is different.

Before changing the class and before going onto more bespoke issues, you should deal with some of the common issues in each class. Some common issues are:

- Commas and other punctuation, such as denomination signs in columns that should be numeric.
- Inconsistent formatting of dates, such as 'December' and also 'Dec' and '12'.
- Unexpected characters, especially in unicode, which may not display consistently.

Typically, we want to fix anything immediately obvious. For instance, remove commas that have been used to group digits in currencies. However, the situation will typically quickly become dire. What we need to do is to look at the membership of each group, and then triage what we will fix. We should probably make the decision of how to triage based on what is likely to have the largest impact. That usually means starting with the counts, sorting in descending order, and then dealing with each as they come.

When the tests of membership are passed, then finally change the class, and run all the tests again. We're adapting this idea from the software development approach of unit testing. Tests are crucial because they enable us to understand whether software (or in this case data) is fit for purpose (Wilson, 2021), Chapter 'Testing'.

Let's run through an example with a collection of strings, some of which are

slightly wrong. This type of output is typical of OCR, which often gets most of the way there, but not quite.

```
messy_string <- c('Rohan, Rhan, ROhan, Roham, ROhan, Rohan, Rohan, ROhan, Rohan , 50han')
messy_string
#> [1] "Rohan, Rhan, ROhan, Roham, ROhan, Rohan, Rohan, ROhan, Rohan , 50han"
```

As before, we first want to get this into a rectangular dataset.

```
my_data <-
 tibble(names = messy_string) %>%
 tidyrr::separate_rows(names, sep = ", ")
my_data
#> # A tibble: 10 x 1
#> names
#> <chr>
#> 1 "Rohan"
#> 2 "Rhan"
#> 3 "ROhan"
#> 4 "Roham"
#> 5 "ROhan"
#> 6 "Rohan"
#> 7 "Rohan"
#> 8 "ROhan"
#> 9 "Rohan "
#> 10 "50han"
```

We now need decide which of these errors we are going to fix. To help us decide which are most important, we'll create a count.

```
my_data %>%
 count(names, sort = TRUE)
#> # A tibble: 7 x 2
#> names n
#> <chr> <int>
#> 1 "Rohan" 3
#> 2 "ROhan" 2
#> 3 "50han" 1
#> 4 "ROhan" 1
#> 5 "Rhan" 1
#> 6 "Roham" 1
#> 7 "Rohan " 1
```

The most common element is the correct one, which is great. The next one -

‘ROhan’ - looks like the ‘o’ has been incorrectly capitalized, and the one after that - ‘5Ohan’ - is distinguished by the ‘5’ instead of an ‘R’. Let’s quickly fix these issues and then redo the count.

```
my_data <-
 my_data %>%
 mutate(names = str_replace_all(names, 'ROhan', 'Rohan'),
 names = str_replace_all(names, '5Ohan', 'Rohan')
)

my_data %>%
 count(names, sort = TRUE)
#> # A tibble: 5 x 2
#> names n
#> <chr> <int>
#> 1 "Rohan" 6
#> 2 "R0han" 1
#> 3 "Rhan" 1
#> 4 "Roham" 1
#> 5 "Rohan " 1
```

Already this is much better and 60 per cent of the values are correct, compared with earlier where it was 30 per cent. There are two more obvious errors - ‘Rhan’ and ‘Roham’ - with the first missing an ‘o’ and the second having an ‘m’ where the ‘n’ should be. Again, we can quickly update and fix those.

```
my_data <-
 my_data %>%
 mutate(names = str_replace_all(names, 'Rhan', 'Rohan'),
 names = str_replace_all(names, 'Roham', 'Rohan')
)

my_data %>%
 count(names, sort = TRUE)
#> # A tibble: 3 x 2
#> names n
#> <chr> <int>
#> 1 "Rohan" 8
#> 2 "R0han" 1
#> 3 "Rohan " 1
```

We have achieved an 80 per cent fix with not too much effort. The two remaining issues are more subtle. The first - ‘R0han’ - has occurred because the ‘o’ has been incorrectly coded as an ‘0’. In some fonts this will show up, but in others it will be more difficult to see. This is a common issue, especially

with OCR, and something to be aware of. The second - ‘Rohan’ - is similarly subtle and is occurring because there is a trailing space. Again, trailing and leading spaces are a common issue. After we fix these two remaining issues then we will have all entries corrected.

```
my_data <-
 my_data %>%
 mutate(names = str_replace_all(names, 'RØhan', 'Rohan'),
 names = stringr::str_trim(names, side = c("right")))
)

my_data %>%
 count(names, sort = TRUE)
#> # A tibble: 1 × 2
#> names n
#> <chr> <int>
#> 1 Rohan 10
```

I’ve been doing the tests in my head in this example - I know that we’re hoping for ‘Rohan’. But we can start to document this test as well. One way is to look to see if values other than ‘Rohan’ exist in the dataset.

```
check_me <-
 my_data %>%
 filter(names != "Rohan")

if (nrow(check_me) > 0) {
 print("Still have values that are not Rohan!")
}
```

### 11.2.5 Iterate, generalize and update

We could now iterate the plan. In this most recent case we started with 10 entries. There is no reason that we couldn’t increase this to 100 or even 1,000. We may need to generalize the cleaning procedures and tests. But eventually we would start to bring the dataset into some sort of order.

---

## 11.3 Case study - Kenya census

To make this all more clear, let’s return to the Kenyan census that we downloaded as PDFs in Chapter 8.

The distribution of population by age, sex, and administrative unit from the 2019 Kenyan census can be downloaded here: <https://www.knbs.or.ke/?wpdmpro=2019-kenya-population-and-housing-census-volume-iii-distribution-of-population-by-age-sex-and-administrative-units>.

And while it is great that they make it easily available, and it is easy to look-up a particular result, it is not overly useful to do larger-scale data analysis, such as building a Bayesian hierarchical model.

In this section we will convert a PDF of Kenyan census results of counts, by age and sex, by county and sub-county, into a tidy dataset that can be analysed. I will draw on and introduce a bunch of handy packages including: `janitor` by [Firke \(2020\)](#), `pdftools` by [Ooms \(2019b\)](#), `tidyverse` by [Wickham et al. \(2019b\)](#), and `stringi` by [Gagolewski \(2020\)](#).

### 11.3.1 Set-up

To get started I need to load the necessary packages.

```
library(janitor)
library(pdftools)
library(tidyverse)
library(stringi)
```

And then I need to read in the PDF that I want to convert.

```
Read in the PDF
all_content <- pdftools::pdf_text("inputs/pdfs/2019_Kenya_census.pdf")
```

The `pdf_text` function from `pdftools` is useful when you have a PDF and you want to read the content into R. For many recently produced PDFs it'll work pretty well, but there are alternatives. If the PDF is an image, then it won't work, and you'll need to turn to OCR.

You can see a page of the PDF here:

```
knitr::include_graphics("figures/2020-04-10-screenshot-of-census.png")
```

2019 Kenya Population and Housing Census: Volume III									
Table 2.3: Distribution of Population by Age, Sex*, County and Sub- County									
MOMBASA		Age	Male	Female	Total	Age	Male	Female	Total
	Total	610,257	598,046	1,208,303	51	4,139	2,577	6,716	
0	15,111	15,009	30,120	52	3,674	2,567	6,241		
1	15,805	15,308	31,113	53	2,788	1,911	4,699		
2	15,088	14,837	29,925	54	3,227	2,117	5,344		
3	14,660	14,031	28,691	55-54	19,855	13,602	33,457		
4	14,061	13,993	28,054	55	3,158	2,059	5,217		
5	74,725	73,178	147,903	56	3,259	2,366	5,625		
6	13,851	14,023	27,874	57	2,881	2,099	4,980		
7	12,889	13,216	26,105	58	2,170	1,769	3,939		
8	13,268	13,203	26,471	59	2,403	1,908	4,311		
9	11,996	12,423	24,419	60	2,538	1,894	4,432		
10	12,523	12,594	25,117	61	1,784	1,444	3,238		
11	12,349	12,343	24,692	62	1,617	1,274	2,891		
12	11,115	11,497	22,612	63	1,462	1,273	2,735		
13	11,661	11,630	23,291	64	1,238	1,038	2,276		
14	11,631	12,081	23,712	65	8,649	7,385	16,034		
15	10,388	10,729	21,117	66	1,555	1,434	2,989		
16	57,144	58,280	115,424	67	1,007	856	1,863		
17	9,199	9,724	18,923	68	1,229	994	2,223		
18	8,922	9,863	18,785	69	721	682	1,403		
19	9,863	10,487	20,350	70	879	886	1,765		
20	8,964	10,217	19,181	71	5,391	4,852	10,243		
21	10,596	12,898	23,494	72	1,027	1,265	2,292		
22	47,544	53,189	100,733	73	624	618	1,242		
23	11,186	14,416	25,602	74	574	518	1,092		
24	11,527	13,706	25,233	75	464	446	910		
25	12,833	15,192	28,025	76	387	472	859		
26	14,195	16,513	30,700	77	3,076	3,319	6,395		
27	14,081	16,084	30,165	78	430	500	930		
28	63,822	75,911	139,733	79	280	315	595		
29	16,231	16,675	32,906	80	263	283	526		
30	14,231	15,107	29,333	81	222	221	443		
31	14,111	14,768	28,879	82	241	310	551		
32	12,333	13,049	25,382	83	75-79	1,436	1,609	3,045	
33	12,921	13,202	26,123	84	80	234	493	727	
34	15,982	16,319	32,301	85	130	178	308		
35	10,866	10,548	21,414	86	138	156	294		
36	13,578	13,379	26,957	87	128	144	272		
37	10,592	10,562	21,154	88	92	165	257		
38	40,098	42,035	89,113	89	80-84	722	1,136	1,858	
39	8,106	7,005	15,111	90	39	107	146		
40	47,078	42,035	89,113	91	17	38	55		
41	10,190	8,555	18,745	92	19	26	45		
42	8,490	6,541	15,031	93	12	24	36		
43	8,405	6,505	14,910	94	11	30	41		
44	6,979	5,521	12,500	95-94	98	225	323		
45	6,034	4,598	10,632	95	11	41	52		
46	40,098	31,720	71,818	96	13	17	30		
47	8,759	6,234	14,993	97	9	12	21		
48	5,852	4,342	10,194	98	8	7	15		
49	6,402	4,282	10,684	99	13	21	34		
50	5,045	3,338	8,383	100+	14	59	73		
51	30,665	21,602	52,267	Not Stated	14	12	26		

\*Intersex population is excluded from the table since it is too small to be distributed by age

18

### 11.3.2 Extract

The first challenge is to get the dataset into a format that we can more easily manipulate. The way that I am going to do this is to consider each page of the PDF and extract the relevant parts. To do this, I first write a function that I want to apply to each page.

```
The function is going to take an input of a page
get_data <- function(i){
```

```

Just look at the page of interest
Based on https://stackoverflow.com/questions/47793326/tabulate-function-in-r
just_page_i <- stringi::stri_split_lines(all_content[[i]])[[1]]

Grab the name of the location
area <- just_page_i[3] %>% str_squish()
area <- str_to_title(area)

Grab the type of table
type_of_table <- just_page_i[2] %>% str_squish()

Get rid of the top matter
just_page_i_no_header <- just_page_i[5:length(just_page_i)] # Just manually for now, but could c

Get rid of the bottom matter
just_page_i_no_header_no_footer <- just_page_i_no_header[1:62] # Just manually for now, but could

Convert into a tibble
demography_data <- tibble(all = just_page_i_no_header_no_footer)

Split columns
demography_data <-
 demography_data %>%
 mutate(all = str_squish(all)) %>% # Any space more than two spaces is squished down to one
 mutate(all = str_replace(all, "10 -14", "10-14")) %>%
 mutate(all = str_replace(all, "Not Stated", "NotStated")) %>% # Any space more than two spaces
 separate(col = all,
 into = c("age", "male", "female", "total", "age_2", "male_2", "female_2", "total_2"),
 sep = " ", # Just looking for a space. Seems to work fine because the tables are pret
 remove = TRUE,
 fill = "right"
)

They are side by side at the moment, need to append to bottom
demography_data_long <-
 rbind(demography_data %>% select(age, male, female, total),
 demography_data %>%
 select(age_2, male_2, female_2, total_2) %>%
 rename(age = age_2, male = male_2, female = female_2, total = total_2)
)

There is one row of NAs, so remove it
demography_data_long <-
 demography_data_long %>%

```

```

janitor::remove_empty(which = c("rows"))

Add the area and the page
demography_data_long$area <- area
demography_data_long$table <- type_of_table
demography_data_long$page <- i

rm(just_page_i,
 i,
 area,
 type_of_table,
 just_page_i_no_header,
 just_page_i_no_header_no_footer,
 demography_data)

return(demography_data_long)
}

```

At this point, I have a function that does what I need to each page of the PDF. I'm going to use the function `map_dfr` from the `purrr` package to apply that function to each page, and then combine all the outputs into one tibble.

```

Run through each relevant page and get the data
pages <- c(30:513)
all_tables <- map_dfr(pages, get_data)
rm(pages, get_data, all_content)

```

### 11.3.3 Clean

I now need to clean the dataset to make it useful.

#### 11.3.3.1 Values

The first step is to make the numbers into actual numbers, rather than characters. Before I can convert the type I need to remove anything that is not a number otherwise it'll be converted into an NA. I first identify any values that are not numbers so that I can remove them.

```

Need to convert male, female, and total to integers
First find the characters that should not be in there
all_tables %>%
 select(male, female, total) %>%
 mutate_all(~str_remove_all(., "[[:digit:]]"))

```

```

 mutate_all(~str_remove_all(., ",")) %>%
 mutate_all(~str_remove_all(., "_")) %>%
 mutate_all(~str_remove_all(., "-")) %>%
 distinct()
#> # A tibble: 39 x 3
#> male female total
#> <chr> <chr> <chr>
#> 1 <NA> <NA> <NA>
#> 2 ":" "Distribution" "of"
#> 3 "Male" "Female" "Total"
#> 4 "" "" ""
#> 5 "by" "Age" "Sex*"
#> 6 "LUNGA" <NA> <NA>
#> 7 "NORTH" <NA> <NA>
#> 8 "SOUTH" <NA> <NA>
#> 9 "RIVER" <NA> <NA>
#> 10 "DELTA" <NA> <NA>
#> # ... with 29 more rows
We clearly need to remove ",", "_", and "-".
This also highlights a few issues on p. 185 that need to be manually adjusted
https://twitter.com/RohanAlexander/status/1244337583016022018
all_tables$male[all_tables$male == "23-Jun"] <- 4923
all_tables$male[all_tables$male == "15-Aug"] <- 4611

```

While you could use the `janitor` package here, it is worthwhile at least first looking at what is going on because sometimes there is odd stuff that `janitor` (and other packages) will deal with, but not in a way that you want. In this case, they've used Excel or similar and this has converted a couple of their entries into dates. If we just took the numbers from the column then we'd have 23 and 15 here, but by inspecting the column we can use Excel to reverse the process and enter the correct values of 4,923 and 4,611, respectively.

Having identified everything that needs to be removed, we can do the actual removal and convert our character column of numbers to integers.

```

all_tables <-
 all_tables %>%
 mutate_at(vars(male, female, total), ~str_remove_all(., ",")) %>% # First get rid of commas
 mutate_at(vars(male, female, total), ~str_replace(., "_", "0")) %>%
 mutate_at(vars(male, female, total), ~str_replace(., "-", "0")) %>%
 mutate_at(vars(male, female, total), ~as.integer())

```

### 11.3.3.2 Areas

The next thing to clean is the areas. We know that there are 47 counties in Kenya, and a whole bunch of sub-counties. They give us a list on pages 19 to 22 of the PDF (document pages 7 to 10). However, this list is not complete, and there are a few minor issues that we'll deal with later.

In any case, I first need to fix a few inconsistencies.

```
Fix some area names
all_tables$area[all_tables$area == "Taita/ Taveta"] <- "Taita/Taveta"
all_tables$area[all_tables$area == "Elgeyo/ Marakwet"] <- "Elgeyo/Marakwet"
all_tables$area[all_tables$area == "Nairobi City"] <- "Nairobi"
```

Kenya has 47 counties, each of which has sub-counties. The PDF has them arranged as the county data then the sub-counties, without designating which is which. We can use the names, to a certain extent, but in a handful of cases, there is a sub-county that has the same name as a county so we need to first fix that.

The PDF is made-up of three tables.

```
all_tables$table %>% table()
#> .
#>
#> 59185
```

So I can first get the names of the counties based on those first two tables and then reconcile them to get a list of the counties.

```
Get a list of the counties
list_counties <-
 all_tables %>%
 filter(table %in% c("Table 2.4a: Distribution of Rural Population by Age, Sex* and County",
 "Table 2.4b: Distribution of Urban Population by Age, Sex* and County"))
) %>%
 select(area) %>%
 distinct()
```

As I hoped, there are 47 of them. But before I can add a flag based on those names, I need to deal with the sub-counties that share their name. We will do this based on the page, then looking it up and deciding which is the county page and which is the sub-county page.

```
The following have the issue of the name being used for both a county and a sub-county:
all_tables %>%
 filter(table == "Table 2.3: Distribution of Population by Age, Sex*, County and Sub- County") %>%
 filter(area %in% c("Busia",
 "Garissa",
 "Homa Bay",
 "Isiolo",
 "Kiambu",
 "Machakos",
 "Makueni",
 "Samburu",
 "Siaya",
 "Tana River",
 "Vihiga",
 "West Pokot"))
) %>%
 select(area, page) %>%
 distinct()
#> # A tibble: 0 x 2
#> # ... with 2 variables: area <chr>, page <int>
```

Now we can add the flag for whether the area is a county and adjust for the ones that are troublesome,

```
Add flag for whether it is a county or a sub-county
all_tables <-
 all_tables %>%
 mutate(area_type = if_else(area %in% list_counties$area, "county", "sub-county"))
Fix the flag for the ones that have their names used twice
all_tables <-
 all_tables %>%
 mutate(area_type = case_when(
 area == "Samburu" & page == 42 ~ "sub-county",
 area == "Tana River" & page == 56 ~ "sub-county",
 area == "Garissa" & page == 69 ~ "sub-county",
 area == "Isiolo" & page == 100 ~ "sub-county",
 area == "Machakos" & page == 154 ~ "sub-county",
 area == "Makueni" & page == 164 ~ "sub-county",
 area == "Kiambu" & page == 213 ~ "sub-county",
 area == "West Pokot" & page == 233 ~ "sub-county",
 area == "Vihiga" & page == 333 ~ "sub-county",
 area == "Busia" & page == 353 ~ "sub-county",
 area == "Siaya" & page == 360 ~ "sub-county",
```

```

area == "Homa Bay" & page == 375 ~ "sub-county",
TRUE ~ area_type
)
)

rm(list_counties)

```

### 11.3.3.3 Ages

Now we can deal with the ages.

First we need to fix some errors.

```

Clean up ages
table(all_tables$age) %>% head()
#>
#> 0 0-4 1 10 10-14
#> 392 484 484 484 484 482
unique(all_tables$age) %>% head()
#> [1] "" "Table" "MOMBASA" "Age" "Total" "0"
Looks like there should be 484, so need to follow up on some:
all_tables$age[all_tables$age == "NotStated"] <- "Not Stated"
all_tables$age[all_tables$age == "43594"] <- "5-9"
all_tables$age[all_tables$age == "43752"] <- "10-14"
all_tables$age[all_tables$age == "9-14"] <- "5-9"
all_tables$age[all_tables$age == "10-19"] <- "10-14"

```

The census has done some of the work of putting together age-groups for us, but we want to make it easy to just focus on the counts by single-year-age. As such I'll add a flag as to the type of age it is: an age group, such as ages 0 to 5, or a single age, such as 1.

```

Add a flag as to whether it's a summary or not
all_tables$age_type <- if_else(str_detect(all_tables$age, c("-")), "age-group", "single-year")
all_tables$age_type <- if_else(str_detect(all_tables$age, c("Total")), "age-group", all_tables$age)

```

At the moment, age is a character variable. We have a decision to make here, because we don't want it to be a character variable (because it won't graph properly), but we don't want it to be a numeric, because there is total and also 100+ in there. For now, we'll just make it into a factor, and at least that will be able to be nicely graphed.

```
all_tables$age <- as_factor(all_tables$age)
```

### 11.3.4 Check

#### 11.3.4.1 Gender sum

Given the format of the data, at this point it is easy to check that `total` is the sum of `male` and `female`.

```
Check the parts and the sums
follow_up <-
 all_tables %>%
 mutate(check_sum = male + female,
 totals_match = if_else(total == check_sum, 1, 0)
) %>%
 filter(totals_match == 0)
```

There is just one that seems wrong.

```
There is just one that looks wrong
all_tables$male[all_tables$age == "10" & all_tables$page == 187] <- as.integer(1)

rm(follow_up)
```

#### 11.3.4.2 Rural-urban split

The census provides different tables for the total of each county and sub-county; and then within each county, for the number in an urban area in that county, and the number in a urban area in that county. Some counties only have an urban count, but we'd like to make sure that the sum of rural and urban counts equals the total count. This requires reshaping the data from a long to wide format.

First, construct different tables for each of the three. I just do it manually, but I could probably do this a nicer way.

```
Table 2.3
table_2_3 <- all_tables %>%
 filter(table == "Table 2.3: Distribution of Population by Age, Sex*, County and Sub- County")
table_2_4a <- all_tables %>%
 filter(table == "Table 2.4a: Distribution of Rural Population by Age, Sex* and County")
table_2_4b <- all_tables %>%
 filter(table == "Table 2.4b: Distribution of Urban Population by Age, Sex* and County")
```

Having constructed the constituent parts, I now join them based on age, area, and whether it is a county.

```
both_2_4s <- full_join(table_2_4a, table_2_4b, by = c("age", "area", "area_type"), suffix = c("_ru"))

all <- full_join(table_2_3, both_2_4s, by = c("age", "area", "area_type"), suffix = c("_all", "_"))

all <-
 all %>%
 mutate(page = glue::glue('Total from p. {page}, rural from p. {page_rural}, urban from p. {page_u',
 select(-page, -page_rural, -page_urban,
 -table, -table_rural, -table_urban,
 -age_type_rural, -age_type_urban
)

rm(both_2_4s, table_2_3, table_2_4a, table_2_4b)
```

We can now check that the sum of rural and urban is the same as the total.

```
Check that the urban + rural = total
follow_up <-
 all %>%
 mutate(total_from_bits = total_rural + total_urban,
 check_total_is_rural_plus_urban = if_else(total == total_from_bits, 1, 0),
 total_from_bits - total) %>%
 filter(check_total_is_rural_plus_urban == 0)

head(follow_up)
#> # A tibble: 0 x 16
#> # ... with 16 variables: age <fct>, male <int>, female <int>, total <int>,
#> # area <chr>, area_type <chr>, age_type <chr>, male_rural <int>,
#> # female_rural <int>, total_rural <int>, male_urban <int>,
#> # female_urban <int>, total_urban <int>, total_from_bits <int>,
#> # check_total_is_rural_plus_urban <dbl>, total_from_bits - total <int>
rm(follow_up)
```

There are just a few, but they only have a difference of 1, so I'll just move on.

#### 11.3.4.3 Ages sum to age-groups

Finally, I want to check that the single age counts sum to the age-groups.

```
One last thing to check is that the ages sum to their age-groups.
follow_up <-
```

```

all %>%
 mutate(groups = case_when(age %in% c("0", "1", "2", "3", "4", "0-4") ~ "0-4",
 age %in% c("5", "6", "7", "8", "9", "5-9") ~ "5-9",
 age %in% c("10", "11", "12", "13", "14", "10-14") ~ "10-14",
 age %in% c("15", "16", "17", "18", "19", "15-19") ~ "15-19",
 age %in% c("20", "21", "22", "23", "24", "20-24") ~ "20-24",
 age %in% c("25", "26", "27", "28", "29", "25-29") ~ "25-29",
 age %in% c("30", "31", "32", "33", "34", "30-34") ~ "30-34",
 age %in% c("35", "36", "37", "38", "39", "35-39") ~ "35-39",
 age %in% c("40", "41", "42", "43", "44", "40-44") ~ "40-44",
 age %in% c("45", "46", "47", "48", "49", "45-49") ~ "45-49",
 age %in% c("50", "51", "52", "53", "54", "50-54") ~ "50-54",
 age %in% c("55", "56", "57", "58", "59", "55-59") ~ "55-59",
 age %in% c("60", "61", "62", "63", "64", "60-64") ~ "60-64",
 age %in% c("65", "66", "67", "68", "69", "65-69") ~ "65-69",
 age %in% c("70", "71", "72", "73", "74", "70-74") ~ "70-74",
 age %in% c("75", "76", "77", "78", "79", "75-79") ~ "75-79",
 age %in% c("80", "81", "82", "83", "84", "80-84") ~ "80-84",
 age %in% c("85", "86", "87", "88", "89", "85-89") ~ "85-89",
 age %in% c("90", "91", "92", "93", "94", "90-94") ~ "90-94",
 age %in% c("95", "96", "97", "98", "99", "95-99") ~ "95-99",
 TRUE ~ "Other"))
) %>%
 group_by(area_type, area, groups) %>%
 mutate(group_sum = sum(total, na.rm = FALSE),
 group_sum = group_sum / 2,
 difference = total - group_sum) %>%
 ungroup() %>%
 filter(age == groups) %>%
 filter(total != group_sum)

head(follow_up)
#> # A tibble: 0 x 16
#> # ... with 16 variables: age <fct>, male <int>, female <int>, total <int>,
#> # area <chr>, area_type <chr>, age_type <chr>, male_rural <int>,
#> # female_rural <int>, total_rural <int>, male_urban <int>,
#> # female_urban <int>, total_urban <int>, groups <chr>, group_sum <dbl>,
#> # difference <dbl>

rm(follow_up)

```

Mt. Kenya Forest, Aberdare Forest, Kakamega Forest are all slightly dodgy. I can't see it in the documentation, but it looks like they have apportioned

these between various countries. It's understandable why they'd do this and it's probably not a big deal, so I'll just move on.

### 11.3.5 Tidy-up

Now that we are confident that everything is looking good, we can just convert it to long-format so that it is easy to work with.

```
all <-
 all %>%
 rename(male_total = male,
 female_total = female,
 total_total = total) %>%
 pivot_longer(cols = c(male_total, female_total, total_total, male_rural, female_rural, total_rural),
 names_to = "type",
 values_to = "number")
) %>%
 separate(col = type, into = c("gender", "part_of_area"), sep = "_") %>%
 select(area, area_type, part_of_area, age, age_type, gender, number)

write_csv(all, path = "outputs/data/cleaned_kenya_2019_census.csv")

head(all)
#> # A tibble: 0 x 7
#> # ... with 7 variables: area <chr>, area_type <chr>, part_of_area <chr>,
#> # age <fct>, age_type <chr>, gender <chr>, number <int>
```

### 11.3.6 Make Monica's dataset

The original purpose of all of this was to make a table for Monica. She needed single-year counts, by gender, for the counties.

```
monicas_dataset <-
 all %>%
 filter(area_type == "county") %>%
 filter(part_of_area == "total") %>%
 filter(age_type == "single-year") %>%
 select(area, age, gender, number)

head(monicas_dataset)
#> # A tibble: 0 x 4
#> # ... with 4 variables: area <chr>, age <fct>, gender <chr>, number <int>
```

```
write_csv(monicas_dataset, "outputs/data/monicas_dataset.csv")
```

I'll leave the fancy stats to Monica, but I'll just make a quick graph of Nairobi.

```
monicas_dataset %>%
filter(area == "Nairobi") %>%
ggplot() +
geom_col(aes(x = age, y = number, fill = gender), position = "dodge") +
scale_y_continuous(labels = scales::comma) +
scale_x_discrete(breaks = c(seq(from = 0, to = 99, by = 5), "100+")) +
theme_classic() +
scale_fill_brewer(palette = "Set1") +
labs(y = "Number",
x = "Age",
fill = "Gender",
title = "Distribution of age and gender in Nairobi in 2019",
caption = "Data source: 2019 Kenya Census")
```

## 11.4 Checks and tests

Robert Caro, the biographer of Lyndon Johnson, spent years tracking down everyone connected to the 36th President of the United States. He went to far as to live in Texas Hill Country for X years so that he could better understand where LBJ was from. When he heard a story that LBJ used to run to the Senate when he worked as a Y, he ran that route multiple times himself to try to understand why LBJ was running. Caro eventually understood it only when he ran the route as the sun was rising, just as LBJ had done, and found that at that time the sun hits the Senate Rotunda and it looks amazing (CITE). This background work enabled him to uncover aspects that no one else knew. For instance, it turns out that LBJ almost surely stole his first election win as a Texas Senator. You need to understand your data to this same extent. Turn every page and go to every extreme.

When we are cleaning data, we are looking for anomalies. We are interested in values that are in there that should not be, but also the opposite situation—values that are missing that should not be. I've talked fairly generally about checks, tests and other considerations. Here I'd like to be more specific about what I mean. There are four tools that you should use to identify these situations: plots, counts, green/red conditions, targets.

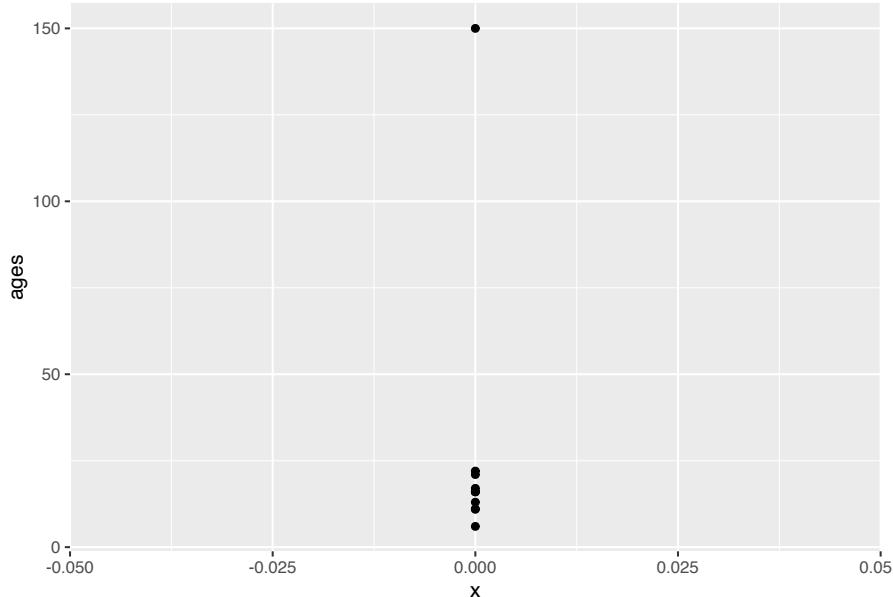
### 11.4.1 Plots

Plots are an invaluable tool when cleaning data, because they show each point in the dataset, in relation to the other points. They are especially useful for identifying when a value doesn't belong. For instance, if a value is expected to be numerical, but it is still a character then it will not plot and a warning will be displayed.

Plots will be especially useful for numerical data, but are still useful for text and categorical data. Let's pretend that we have a situation where we are interested in a person's age, for some youth survey. We have the following data:

```
raw_data <-
 tibble(ages = c(11, 17, 22, 13, 21, 16, 16, 6, 16, 11, 150))

raw_data %>%
 ggplot(aes(y = ages, x = 0)) +
 geom_point()
```



The graph clearly shows the unexpected value of 150. The most likely explanation is that the data were incorrectly entered with a trailing 0, and should be 15.

We can fix that, and document it, and then redo the graph, so see that everything seems more reasonable now.

### 11.4.2 Counts

We want to focus on getting most of the data right. So we are interested in the counts of unique values. Hopefully a majority of the data are concentrated in the most common counts. But it can also be useful to invert it, and see what is especially uncommon. The extent to which we want to deal with these depends on what we need. Ultimately, each time we fix one we are getting very few additional observations, potentially even just one! Counts are especially useful with text or categorical data, but can be helpful with numerical as well.

Let's see an example.

```
raw_data <-
 tibble(country = c('Australie', 'Austrelia', 'Australie', 'Australie', 'Aeustralia', 'Austraia',
)
)

raw_data %>%
 count(country, sort = TRUE)
#> # A tibble: 5 x 2
#> country n
#> <chr> <int>
#> 1 Australia 4
#> 2 Australie 3
#> 3 Aeustralia 1
#> 4 Austraia 1
#> 5 Austrelia 1
```

The use of this count clearly identifies where we should spend our time - changing 'Australie' to 'Australia' would almost double our amount of useable data.

### 11.4.3 Go/no-go

Some things are so important that you require that your cleaned dataset have them. These are go/no-go conditions. They would typically come out of experience, expert knowledge, or the planning and simulation exercises. An example may be that there are no negative numbers in an age column, and no ages above 140.

For these we could specifically require that the condition is met. Other examples include:

- If doing cross-country analysis, then a list of country names that we know should be in our dataset would be useful. Our no-go conditions would then be if there were: 1) values not in that list in our dataset, or, vice versa; 2) countries that we expected to be in there that were not.

To have a concrete example, let's consider if we were doing some analysis about the five largest counties in Kenya: 'Nairobi', 'Kiambu', 'Nakuru', 'Kakamega', 'Bungoma'. Let's create that array first:

```
correct_counties <- c('Nairobi', 'Kiambu', 'Nakuru', 'Kakamega', 'Bungoma')
```

We begin with the following dataset:

```
top_five_kenya <-
 tibble(county = c('Nairobi', 'Nairobi', 'Nakuru', 'Kakamega', 'Nakuru',
 'Kiambu', 'Kiambu', 'Kakamega', 'Bun8oma', 'Bungoma')
)

top_five_kenya %>%
 count(county, sort = TRUE)
#> # A tibble: 9 x 2
#> county n
#> <chr> <int>
#> 1 Nakuru 2
#> 2 Bun8oma 1
#> 3 Bungoma 1
#> 4 Kakamega 1
#> 5 Kakamega 1
#> 6 Kiambu 1
#> 7 Kiambu 1
#> 8 Nairobi 1
#> 9 Nairobi 1
```

Based on the count we know that we have to fix some of them and there are two with numbers that are obvious fixes:

```
top_five_kenya <-
 top_five_kenya %>%
 mutate(county = str_replace_all(county, 'Nairobi', 'Nairobi'),
 county = str_replace_all(county, 'Bun8oma', 'Nairobi')
)

top_five_kenya %>%
 count(county, sort = TRUE)
#> # A tibble: 7 x 2
#> county n
#> <chr> <int>
#> 1 Nairobi 3
#> 2 Nakuru 2
```

```
#> 3 Bungoma 1
#> 4 Kabamega 1
#> 5 Kakamega 1
#> 6 Kiambu 1
#> 7 Kiambu 1
```

At this point we can use our go/no-go conditions to decide whether we are finished or not.

```
top_five_kenya$county %>% unique()
#> [1] "Nairobi" "Nakuru" "Kakamega" "Kiambu" "Kiambu" "Kabamega" "Bungoma"

if(all(top_five_kenya$county %>% unique() == top_five_kenya)) {
 "Oh no"
}
if(all(top_five_kenya==top_five_kenya$county %>% unique())) {
 "Oh no"
}
```

And so it is clear that we still have cleaning to do!

We may also find similar conditions from experts and those with experience in the particular field.

#### 11.4.4 Class

I can't emphasize this enough, but it is vital that you put in place explicit checks of class because getting this wrong can have a large effect on your analysis. In particular:

- check whether some value should be a number or a factor.
- check that dates are correctly formatted.

To understand why it is important to be clear about whether a value is a number or a factor, consider the following situation:

```
some_data <-
 tibble(response = c(1, 1, 0, 1, 0, 1, 1, 0, 0),
 group = c(1, 2, 1, 1, 2, 3, 1, 2, 3))
```

Let's start with `group` as an integer and look at a logistic regression.

```
some_data %>%
 mutate(group = as.integer(group)) %>%
```

```

lm(response~group, data = .) %>%
 summary()
#>
#> Call:
#> lm(formula = response ~ group, data = .)
#>
#> Residuals:
#> Min 1Q Median 3Q Max
#> -0.68 -0.52 0.32 0.32 0.64
#>
#> Coefficients:
#> Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 0.8400 0.4495 1.869 0.104
#> group -0.1600 0.2313 -0.692 0.511
#>
#> Residual standard error: 0.5451 on 7 degrees of freedom
#> Multiple R-squared: 0.064, Adjusted R-squared: -0.06971
#> F-statistic: 0.4786 on 1 and 7 DF, p-value: 0.5113

```

Now we can try it as a factor. The interpretation of the variable is completely different.

```

some_data %>%
 mutate(group = as.factor(group)) %>%
 lm(response~group, data = .) %>%
 summary()
#>
#> Call:
#> lm(formula = response ~ group, data = .)
#>
#> Residuals:
#> Min 1Q Median 3Q Max
#> -0.7500 -0.3333 0.2500 0.2500 0.6667
#>
#> Coefficients:
#> Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 0.7500 0.2826 2.654 0.0378 *
#> group2 -0.4167 0.4317 -0.965 0.3717
#> group3 -0.2500 0.4895 -0.511 0.6278
#> ---
#> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 0.5652 on 6 degrees of freedom

```

```
#> Multiple R-squared: 0.1375, Adjusted R-squared: -0.15
#> F-statistic: 0.4783 on 2 and 6 DF, p-value: 0.6416
```

Another critical aspect is to check the dates. In particular we want to try to make it into the following format: YYYY-MM-DD. There are of course differences of opinion as to what is an appropriate date format in the broader world, and reasonable people can differ on whether 1 July 2010 or July 1, 2020, is better, but YYYY-MM-DD is the format that is generally most appropriate for data.

---

## 11.5 Naming things

---

An improved scanning software we developed identified gene name errors in 30.9% (3,436/11,117) of articles with supplementary Excel gene lists; a figure significantly higher than previously estimated. This is due to gene names being converted not just to dates and floating-point numbers, but also to internal date format (five-digit numbers).

[Abeysooriya et al. \(2021\)](#)

---

<https://neverworkintheory.org/2021/08/09/abbreviated-vs-full-names.html>

Names matter. I wrote this book on land that is today named Toronto, which is within a country named Canada, but for a long time before was known as Turtle Island. While not common, these days people will sometimes still refer to themselves as being on Turtle Island. That tells us something about them, and our use of the name Canada tells them something about us. There is a big rock in the centre of the country that I'm from, Australia. For a long time, it was called Uluru, then it was known as Ayers Rock. Today it has a dual name the combines both, and the choice of which name you use tells someone something about you. Even the British Royal Family recognise the power of names. In 1917 they changed from the House of Saxe-Coburg and Gotha to the House of Windsor, due to a feeling that the former was too Germanic given World War I was ongoing. Names matter in everyday life. And they matter in data science too.

The importance of names, and of ignoring existing claims through re-naming was clear in those cases, but we see it in data science as well. We need to be very careful when we name our datasets, our variables, and our functions. There is a tendency, these days, to call the variable ‘gender’ even though it may only have male and female, because we do not want to say the word ‘sex’. (Tukey, 1962) essentially defines what we today call data science, but it was popularised by folks in computer science in the 2010s who ignored, either deliberately or through ignorance, what came before them. The past ten years has been characteristic by the renaming of concepts that were well-established in the fields that computer science has recently expanded into. For instance, the use of binary variables in regression, sometimes called ‘dummy variables’, is called one-hot encoding in computer science. Like all fashions, this one will pass also. We most recently saw this through the 1980s through to early 2010s with economics. Economists described themselves as the ‘queen of the social sciences’ and self-described as imperialistic (Lazear, 2000). We are now recognising the costs of this imperialism in social sciences, and in the future we will look back and count the cost of computer science imperialism in data science. The key here is that no area of study is ever *terra nullius*, or nobody’s land. It is important to recognise, adopt, and use existing names, and practices.

Names give places meaning, and by ignoring existing names, we ignore what has come before us. Kimmerer (2012, p. 34) describes how ‘Tahawus is the Algonquin name for Mount Marcy, the highest peak in the Adirondacks. It’s called Mount March to commemorate a governor who never set foot on those wild slopes.’ She continues that ‘[w]hen we call a place by name it is transformed from wilderness to homeland.’ She is talking with regard to physical places, but the same is true of our function names, our variable names and our dataset names. When we use gender instead of sex because we don’t want to say sex in front of others, we ignore the preferences of those that provided data.

In addition to respecting the nature of the data, names need to satisfy two additional considerations: 1) they need to be machine readable, and 2) they need to be human readable.

Machine readable names is an easier standard to meet, but usually means avoiding spaces and special characters. A space can be replaced with a under-bar. Usually, special characters should just be removed because they can be inconsistent between different computers and languages. The names should also be unique within a dataset, and unique within a collection of datasets unless that particular column is being deliberately used as a key to join different datasets.

An especially useful function to use to get closer to machine readable names is `janitor::clean_names()` which is from the `janitor` package (Firke, 2020).

This deals with those issues mentioned above as well as a few others. We can see an example.

```
bad_names_good_names <-
 tibble(
 'First' = c(1),
 'second name has spaces' = c(1),
 'weird#symbol' = c(1),
 'InCoNsIsTaNtCaPs' = c(1)
)

bad_names_good_names
#> # A tibble: 1 × 4
#> First `second name has spaces` `weird#symbol` InCoNsIsTaNtCaPs
#> <dbl> <dbl> <dbl> <dbl>
#> 1 1 1 1 1

bad_names_good_names <-
 bad_names_good_names %>%
 janitor::clean_names()

bad_names_good_names
#> # A tibble: 1 × 4
#> first second_name_has_spaces weird_number_symbol in_co_ns_is_ta_nt_ca_ps
#> <dbl> <dbl> <dbl> <dbl>
#> 1 1 1 1 1
```

Human readable names require an additional layer! You need to consider other cultures and how they may interpret some of the names that you're using. You also need to consider different experience levels that subsequent users of your dataset may have. This is both in terms of experience with programming and statistics, but also experience with similar datasets. For instance, a column of 'flag' is often used to signal that a column contains data that needs to be followed up with or treated carefully in some way. An experienced analyst will know this, but a beginner will not. Try to use meaningful names wherever possible (Lin et al., 2020).

One interesting feature of R is that in certain cases partial matching on names is possible. For instance:

```
never_use_partial_matching <-
 data.frame(
 my_first_name = c(1, 2),
 another_name = c("wow", "great")
)
```

```
never_use_partial_matching$my_first_name
#> [1] 1 2
never_use_partial_matching$my
#> [1] 1 2
```

This behaviour is not possible within the `tidyverse` (for instance if `data.frame` were replaced with `tibble` in the above code) and I recommend never using this feature. It makes it more difficult to understand your code after a break, and for others to come to it fresh.

---

## 11.6 Exercises and tutorial

### 11.6.1 Exercises

1. Is the following an example of tidy data?

```
tibble(name = c('Anne', 'Bethany', 'Stephen', 'William'),
 age_group = c('18-29', '30-44', '45-60', '60+'),
)
#> # A tibble: 4 x 2
#> name age_group
#> <chr> <chr>
#> 1 Anne 18-29
#> 2 Bethany 30-44
#> 3 Stephen 45-60
#> 4 William 60+
```

2. If I am dealing with ages then what is the most likely class for the variable? [Select all that apply.]
  - a. integer
  - b. matrix
  - c. numeric
  - d. factor

### 11.6.2 Tutorial

With regard to [Jordan \(2019\)](#), [D'Ignazio and Klein \(2020\)](#), Chapter 6, [Au \(2020\)](#), and other relevant work, to what extent do you think we should let the data speak for themselves? [Please write a page or two.]



# 12

---

## *Storing and retrieving data*

---

**STATUS:** Under construction.

**Required reading**

**Recommended reading**

**Key concepts/skills/etc**

- **Key libraries**
  - **Key functions/etc**
  -
- 

### 12.1 Introduction

After you've put together a dataset, an important part of being responsible is storing it appropriately and enabling easy retrieval. While it is certainly possible to be especially concerned about this, and entire careers are based on the storage and retrieval of data, to a certain extent, the baseline here is not onerous. If you can get it off your own computer then you are half-way there! Confirming that someone else can retrieve it and use it, puts you much further than most.

That said, the FAIR principles are especially useful to be more formal about data management. These are ([Wilkinson et al., 2016](#)):

1. Findable. This means that there is one, unchanging, identifier for the dataset and the dataset has high-quality descriptions and explanations.
2. Accessible.
3. Interoperable.
4. Reusable.

It's important to recognise that just because a dataset is FAIR, it is not necessarily an unbiased representation of the world.

---

**Oh, you think we have good data on that!** One representation of reality that is commonplace is in chess. A chess board (see Figure X - add photo of a chess board) is a 8 x 8 board of alternating black and white squares. The squares are denoted by a unique combination of a letter (A-G) and a number (1-8). And each piece has a unique abbreviation, for instance pawns are X, and knights are Y. A game is recorded by each player noting the move. In this way the entire game can be recreated. The 2021 World Championship was contested by Magnus Carlsen and Ian Nepomniachtchi. Figure X shows a score sheet from Game 6. There were a variety of reasons this game was particularly noteworthy, but one the uncharacteristic mistakes that both Carlsen and Nepomniachtchi made. For instance, at Move 32 Carlsen did not exploit an opportunity; and Move 36 a different move would have provided Nepomniachtchi with a promising endgame (CITATION). One reason for this may have been that both players at that point in the game had very little time remaining—they had to decide on their moves very quickly. But there is no sense of that in the representation provided by the game sheet. It is a ‘correct’ representation of what happened in the game, but not necessarily why it happened.

---

---

## 12.2 Plan

Michener (2015)

Information Science and libraries

Hart et al. (2016)

## 12.3 R Packages for data

---

### 12.4 Documentation

Datasheets ([Gebru et al., 2020](#)) are an increasingly critical aspect of data science. Datasheets are basically nutrition labels for datasets. The process of creating them enables us to think more carefully about what we will feed our model. More importantly, they enable others to better understand what we fed our model. Recently researchers went back and wrote a datasheet for one of the most popular datasets in computer science, and they found that around 30 per cent of the data were duplicated ([Bandy and Vincent, 2021](#)).

Instead of telling you how unhealthy various foods are, a datasheet tells you things like:

- ‘Who created the dataset and on behalf of which entity?’
- ‘Who funded the creation of the dataset?’
- Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?’
- ‘Is any information missing from individual instances?’

If you have done a lot of work to create the dataset that you analyze, then it may make sense to try to publish and share it on its own. But typically a datasheet might live in an appendix to the main work.

---

## 12.5 Exercises and tutorial

### 12.5.1 Exercises

1. According to [Gebru et al. \(2020, p.2\)](#), a datasheet should document a dataset’s (please select all that apply):
  - a. composition.
  - b. recommended uses.
  - c. motivation.
  - d. collection process.
2. Following [Wilkinson et al. \(2016\)](#), which of the following are FAIR principles (please select all that apply)?
  - a. Findable.
  - b. Approachable.
  - c. Interoperable.

- d. Reusable.
- e. Integrated.
- f. Fungible.
- g. Reduced.
- h. Accessible.

### **12.5.2 Tutorial**

Look into how IQ tests are conducted and what goes into them. To what extent do you think they measure intelligence? Some aspects that you may like to think about in answering that question include: Who decides what is intelligence? How is this updated? What is missing from that definition? To what extent is this generalisable? You should write a page or two.

# 13

---

## *Disseminating and protecting data*

---

**STATUS:** Under construction.

- Hawes, M. B. (2020). Implementing Differential Privacy: Seven Lessons From the 2020 United States Census. Harvard Data Science Review. <https://doi.org/10.1162/99608f92.353c6f99>
- <https://hdsr.mitpress.mit.edu/pub/g9o4z8au/release/2>
- [https://www.census.gov/newsroom/blogs/research-matters/2020/02/census\\_bureau\\_works.html](https://www.census.gov/newsroom/blogs/research-matters/2020/02/census_bureau_works.html)

**Required reading**

**Recommended reading**

**Key concepts/skills/etc**

- - **Key libraries**
  - 
  - **Key functions/etc**
  -
- 

### 13.1 Introduction

---

### 13.2 What is personally identifying information

Zook et al. (2017)

---

### 13.3 Hashing and salting

---

### 13.4 GDPR and HIPAA

---

### 13.5 Making fake data to distribute when you can't share the real stuff

---

### 13.6 Differential privacy

Kenny et al. (2021)

Ruggles et al. (2019)

Suriyakumar et al. (2020)

---

### 13.7 Exercises and tutorial

#### 13.7.1 Exercises

#### 13.7.2 Tutorial

---

---

# Part V

# Modelling



# 14

---

## *Exploratory data analysis*

---

**STATUS:** Under construction.

### Required reading

- Baracas, Solon, and Danah Boyd, 2017, ‘Engaging the ethics of data science in practice’, *Communications of the ACM*, 60.11 (2017): 23-25.
- DiCiccio, Thomas J., and Mary E. Thompson, 2004, ‘A Conversation with Donald A. S. Fraser’, *Statistical Science*, 19 (2) pp. 370-386, [https://utstat.toronto.edu/craiu/DonFraser\\_SSInterview.pdf](https://utstat.toronto.edu/craiu/DonFraser_SSInterview.pdf).
- Jordan, Michael I, 2019, ‘AI - The revolution hasn’t started yet’, *Harvard Data Science Review*, 1 July, <https://hdsr.mitpress.mit.edu/pub/wot7mkcl>.
- Tukey, John W., 1961, ‘The Future of Data Analysis’, *The annals of mathematical statistics*, Part 1 ‘General Considerations’, <https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-33/issue-1/The-Future-of-Data-Analysis/10.1214/aoms/1177704711.full>.
- Wickham, Hadley, and Garrett Grolemund, 2017, *R for Data Science*, Chapters 3 and 7, <https://r4ds.had.co.nz/>.

### Recommended reading

- Hall, Megan, 2019, ‘Exploratory Data Analysis Using Tidyverse’, <https://hockey-graphs.com/2019/10/08/exploratory-data-analysis-using-tidyverse/>.
- Kommenda, Niko, Helen Pidd and Libby Brooks, 2020, ‘Revealed: the areas in the UK with one Airbnb for every four homes’, *The Guardian*, 20 February, <https://www.theguardian.com/technology/2020/feb/20/revealed-the-areas-in-the-uk-with-one-airbnb-for-every-four-homes>.
- Silge, Julia, 2018, ‘Understanding PCA using Stack Overflow data’, <https://juliasilge.com/blog/stack-overflow-pca/>.
- Soetewey, Antoine, 2020, ‘Descriptive statistics in R’, <https://www.statsandr.com/blog/descriptive-statistics-in-r/>.
- Stodulka, Jiri, 2019, ‘Toronto Crime and Folium’, <https://www.jiristodulka.com/post/toronto-crime/>.
- Wong, Julia Carrie, 2020, ‘One year inside Trump’s monumental Facebook campaign’, *The Guardian*, 29 January, <https://www.theguardian.com/us-news/2020/jan/28/donald-trump-facebook-ad-campaign-2020-election>.

### Key concepts/skills/etc

- Quickly coming to terms with a new dataset by constructing graphs and tables.
- Understanding the issues and features of the dataset and how this may affect your modelling decisions.
- Thinking about missing values and outliers.

### Key libraries

- `broom`
- `ggrepel`
- `here`
- `janitor`
- `lubridate`
- `opendatatoronto`
- `tidymodels`
- `tidyverse`
- `visdat`

### Key functions/etc

- `augment()`
- `clean_names()`
- `coord_flip()`
- `count()`
- `distinct()`
- `facet_grid()`
- `facet_wrap()`
- `geom_bar()`
- `geom_col()`
- `geom_density()`
- `geom_histogram()`
- `geom_line()`
- `geom_point()`
- `geom_smooth()`
- `geom_text_repel()`
- `get_dupes()`
- `glance()`
- `if_else()`
- `ifelse()`
- `initial_split()`
- `left_join()`
- `mutate()`
- `mutate_all()`
- `names()`
- `ncol()`
- `nrow()`
- `pivot_wider()`

- `scale_color_brewer()`
- `scale_fill_brewer()`
- `scale_x_log10()`
- `scale_y_log10()`
- `str_detect()`
- `str_extract()`
- `str_remove()`
- `str_split()`
- `str_starts()`
- `summarise()`
- `summarise_all()`
- `theme_classic()`
- `theme_minimal()`
- `vis_dat()`
- `vis_miss()`

## Quiz

1. In your own words what is exploratory data analysis (this will be difficult, but please write only one nuanced paragraph)?
2. In Tukey's words, what is exploratory data analysis (please write one paragraph)?
3. Who was Tukey (please write a paragraph or two)?
4. What is Tukey's link to DoSS (hint: he was an advisor on someone's PhD - who was that person)?
5. Can you identify a female equivalent to Tukey who we (as historians of statistics) may have overlooked?
6. If you have a dataset called 'my\_data', which has two columns: 'first\_col' and 'second\_col', then could you please write some rough R code that would generate a graph (the type of graph doesn't matter).
7. Consider a dataset that has 500 rows and 3 columns, so there are 1,500 cells. If 100 of the cells are missing data for at least one of the columns, then would you remove the whole row from your dataset or try to run your analysis on the data as is, or some other procedure? What if your dataset had 10,000 rows instead, but the same number of missing cells?
8. Please note three ways of identifying unusual values.
9. What is the difference between a categorical and continuous variable?
10. What is the difference between a factor and an integer variable?
11. How can we think about who is systematically excluded from a dataset?
12. Using the `opendatatoronto` package, download the data on mayoral campaign contributions for 2014. (note: the 2014 file you will get

from `get_resource`, so just keep the sheet that relates to the Mayor election).

1. Clean up the data format (fixing the parsing issue and standardizing the column names using `janitor`)
2. Summarize the variables in the dataset. Are there missing values, and if so, should we be worried about them? Is every variable in the format it should be? If not, create new variable(s) that are in the right format.
3. Visually explore the distribution of values of the contributions. What contributions are notable outliers? Do they share a similar characteristic(s)? It may be useful to plot the distribution of contributions without these outliers to get a better sense of the majority of the data.
4. List the top five candidates in each of these categories: 1) total contributions; 2) mean contribution; and 3) number of contributions.
5. Repeat that process, but without contributions from the candidates themselves.
6. How many contributors gave money to more than one candidate?
13. Name three geoms that produce graphs that have bars on them `ggplot()`.
14. Consider a dataset 10,000 observations and 27 variables. For each observation, there is at least one missing variable. Please discuss, in a paragraph or two, the steps that you would take to understand what is going on.
15. Known missing data, are those that leave holes in your dataset. But what about data that were never collected? Please look at McClelland, Alexander, 2019, ‘“Lock This Whore Up”: Legal Violence and Flows of Information Precipitating Personal Violence against People Criminalised for HIV-Related Crimes in Canada’, European Journal of Risk Regulation, 10 (1), pp. 132-147. Then look at Policing the Pandemic - <https://www.policingthepandemic.ca/>. Look into how they gathered their dataset and what it took to put this together. What is in the dataset and why? What is missing and why? How could this affect the results? How might similar biases enter into other datasets that you have used or read about?

---

## 14.1 Introduction

---

The future of data analysis can involve great progress, the overcoming of real difficulties, and the provision of a great service to all fields of science and technology. Will it? That remains to us, to our willingness to take up the rocky road of real problems in preference to the smooth road of unreal assumptions, arbitrary criteria, and abstract results without real attachments. Who is for the challenge?

Tukey (1962, p. 64).

---

Exploratory data analysis is never finished, you just die. It is the active process of exploring and becoming familiar with your data. Like a farmer with their hands in the earth, you need to know every contour and aspect of your data. You need to know how it changes, what it shows, hides, and what are its limits. Exploratory data analysis is the unstructured process of doing this.

That said, exploratory data analysis (EDA) is not something that ends up in your final paper. It is a means to an end and while it will inform your entire paper, especially the data section, it's not typically something that belongs in a final draft. The best way to proceed is to make a separate .Rmd and add code and brief notes as you go. Don't delete previous code, just add to it. When you run out of time, you'll have a useful notebook that captures your exploration. This is a document for you and your collaborators and will guide all the subsequent modelling that you do.

EDA draws on everything that you know as an analyst. Every tool is fair game and should be considered. Look at the raw data, make some tables, some plots, some summary statistics, make some models. The key here is to iterate, move quickly not perfectly, and come to understand your data.

In this chapter we will work with real data that has many issues so that you can understand the main characteristics and potential issues. We will use the `opendatatoronto` package (Gelfand, 2020), among other sources. There are a lot of options for EDA (Staniak and Biecek, 2019) and here I focus on a few of them.

---

## 14.2 Case study - TTC subway delays

This section was written with Monica Alexander<sup>1</sup>.

---

<sup>1</sup><https://www.monicaalexander.com/>

### 14.2.1 Introduction

The `opendatatoronto` package (Gelfand, 2020) provides an interface to all data available on the Open Data Portal<sup>2</sup> provided by the City of Toronto. We are going to use that to take a quick look at the subway delays. We're additionally going to especially draw on the `tidyverse` (Wickham et al., 2019a), as well as the `ggrepel` (Slowikowski, 2021), `janitor` (Firke, 2020), `lubridate` (Grolemund and Wickham, 2011), and `visdat` (Tierney, 2017) packages.

```
library(opendatatoronto)
library(tidyverse)
library(ggrepel)
library(janitor)
library(lubridate)
library(visdat)
```

### 14.2.2 Gather the data

To begin with, use `opendatatoronto::list_packages()` to look at some of the datasets that available.

```
all_data <- opendatatoronto::list_packages(limit = 500)
all_data
#> # A tibble: 425 x 11
#> title id topics civic_issues publisher excerpt
#> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 Daily ~ 21c83~ Communi~ Affordable ho~ Shelter,~ "Daily ~
#> 2 Municipi~ 57b22~ Busines~ <NA> Municipa~ "Some b~
#> 3 Early0~ 26196~ Communi~ Poverty reduc~ Children~ "Early0~
#> 4 Chemic~ ae8ee~ Environ~ Climate change Toronto ~ "This d~
#> 5 Apartm~ 4ef82~ Locatio~ Affordable ho~ Municipa~ "This d~
#> 6 Addres~ abedd~ Locatio~ Mobility Informat~ "This d~
#> 7 Proper~ 1aca~ Locatio~ Mobility Informat~ "This d~
#> 8 Lobbyi~ 6a87b~ City go~ <NA> Lobbyist~ "The Lo~
#> 9 Toront~ 1d079~ City go~ Mobility Informat~ "Linear~
#> 10 Buildi~ 8219e~ Develop~ <NA> Toronto ~ "Provid~
#> # ... with 415 more rows, and 5 more variables:
#> # dataset_category <chr>, num_resources <int>,
#> # formats <chr>, refresh_rate <chr>,
#> # last_refreshed <date>
```

We'll download the data on TTC subway delays in 2019. There are multiple

---

<sup>2</sup><https://open.toronto.ca/>

files for 2019 so we need to get them all and then make them into one big dataframe.

```
We know this number based on the 'id' of the interest.
ttc_resources <-
 list_package_resources("996cf8d-fb35-40ce-b569-698d51fc683b")

ttc_resources <-
 ttc_resources %>%
 mutate(year = str_extract(name, "201.?"))

delay_2019_ids <-
 ttc_resources %>%
 filter(year==2019) %>%
 select(id) %>%
 pull()

delay_2019 <- c()

for(i in 1:length(delay_2019_ids)) {
 delay_2019 <- bind_rows(delay_2019, get_resource(delay_2019_ids[i]))
}

make the column names nicer to work with
delay_2019 <- clean_names(delay_2019)
```

Let's also download the delay code and readme, as reference. You'd probably want to save all this into an 'inputs' folder, as this is the raw data that would underpin any analysis.

```
delay_codes <- get_resource("fece136b-224a-412a-b191-8d31eb00491e")
#> New names:
#> * ` ` -> ...
#> * `CODE DESCRIPTION` -> `CODE DESCRIPTION...3`
#> * ` ` -> ...
#> * ` ` -> ...
#> * `CODE DESCRIPTION` -> `CODE DESCRIPTION...7`

delay_data_codebook <- get_resource("54247e39-5a7d-40db-a137-82b2a9ab0708")
```

This dataset has a bunch of interesting variables. You can refer to the readme for descriptions. Our outcome of interest is `min_delay`, which give the delay in minutes.

```
head(delay_2019)
#> # A tibble: 6 x 10
#> date time day station code min_delay
#> <dttm> <chr> <chr> <chr> <dbl>
#> 1 2019-01-01 00:00:00 01:08 Tuesday YORK MI~ PUSI 0
#> 2 2019-01-01 00:00:00 02:14 Tuesday ST ANDR~ PUMST 0
#> 3 2019-01-01 00:00:00 02:16 Tuesday JANE ST~ TUSC 0
#> 4 2019-01-01 00:00:00 02:27 Tuesday BLOOR S~ SUO 0
#> 5 2019-01-01 00:00:00 03:03 Tuesday DUPONT ~ MUATC 11
#> 6 2019-01-01 00:00:00 03:08 Tuesday EGLINTO~ EUATC 11
#> # ... with 4 more variables: min_gap <dbl>, bound <chr>,
#> # line <chr>, vehicle <dbl>
```

Next we're going to highlight some tools that might be useful when you are getting used to a new dataset. There's no one way to explore, but it's important to keep in mind:

- what should your variables look like (type, values, distribution, etc);
- what would be surprising (outliers etc); and
- what is your end goal (here, it might be understanding factors associated with delays, e.g. stations, time of year, time of day, etc).

In any data analysis project, if it turns out you have data issues, surprising values, missing data etc, it's important you document anything you found and the subsequent steps or assumptions you made before moving onto your data analysis and modeling.

As always:

- 1) Start with an end in mind.
- 2) Be as lazy as possible.

### 14.2.3 Sanity Checks

We need to check that the variables are what they say they are. If they aren't, the natural next question is to what to do with any issues. Should we recode them, or even remove them? It's important to distinguish between factors and characters, as well as between factors and numerics.

For instance, have a look at the column that claims to be the days of week using `unique()`.

```
unique(delay_2019$day)
#> [1] "Tuesday" "Wednesday" "Thursday" "Friday"
#> [5] "Saturday" "Sunday" "Monday"
```

Another function that's useful here is `table()`.

```
table(delay_2019$day)
#>
#> Friday Monday Saturday Sunday Thursday Tuesday
#> 2979 2970 2238 1978 3116 2939
#> Wednesday
#> 3002
```

Let's now check the lines.

```
unique(delay_2019$line)
#> [1] "YU" "BD"
#> [3] "YU/BD" "SHP"
#> [5] "SRT" NA
#> [7] "YUS" "B/D"
#> [9] "BD LINE" "999"
#> [11] "YU/ BD" "YU & BD"
#> [13] "BD/YU" "YU\ \BD"
#> [15] "46 MARTIN GROVE" "RT"
#> [17] "BLOOR-DANFORTH" "YU / BD"
#> [19] "134 PROGRESS" "YU - BD"
#> [21] "985 SHEPPARD EAST EXPR" "22 COXWELL"
#> [23] "100 FLEMINGDON PARK" "YU LINE"
```

It looks like we have many issues here, and some of them have an obvious re-code, but some do not. Should we drop them? There's no absolute right answer here, it depends on what you're using the data for, but you need to be aware of these issues in the data.

#### 14.2.4 Missing values

Exploring missing data is a course in itself, but the main point is that their presence (or lack thereof) will haunt your analysis. Insert joke about ghost-busters here.

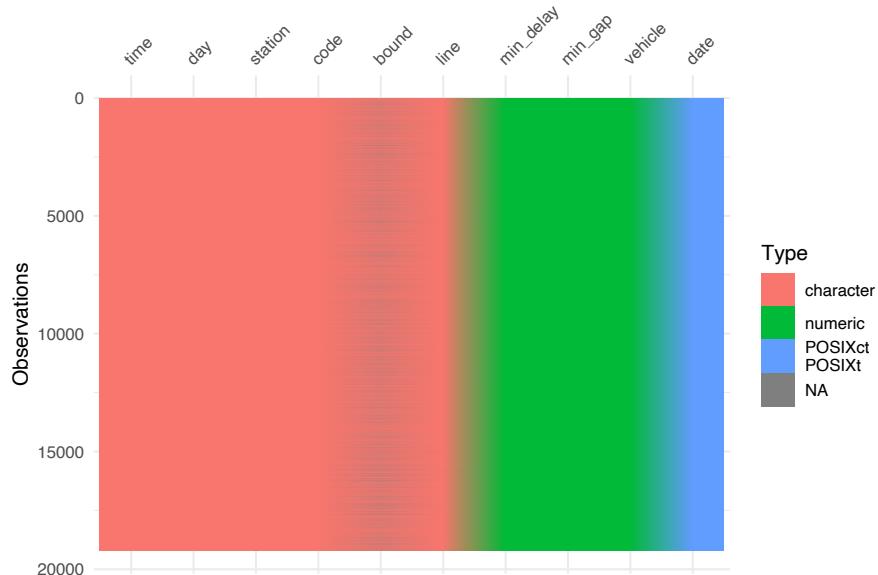
To get started look at known-unknowns, which are the NAs for each variable.

```
delay_2019 %>%
 summarise_all(list(~sum(is.na(.))))
#> # A tibble: 1 x 10
#> date time day station code min_delay min_gap bound
#> <int> <int> <int> <int> <int> <int> <int> <int>
```

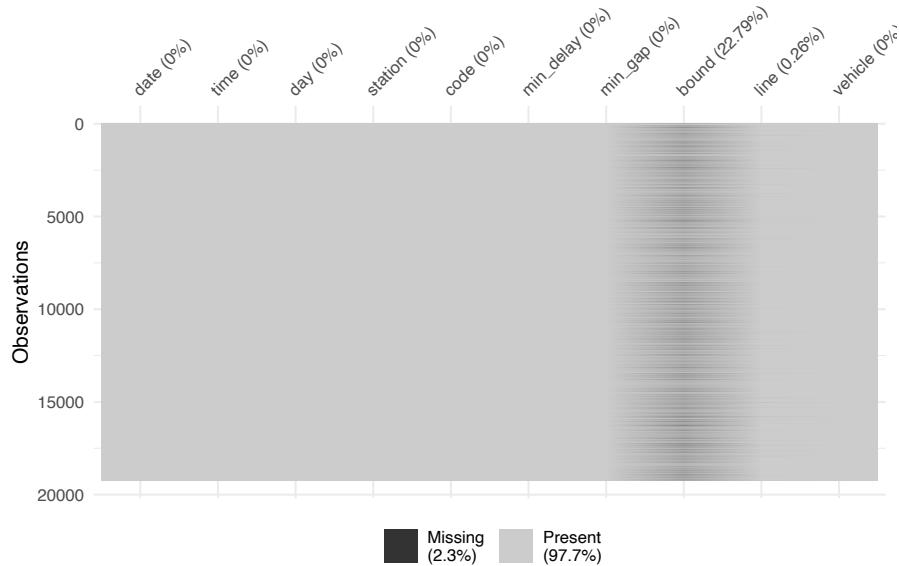
```
#> 1 0 0 0 0 0 0 0 0 4380
#> # ... with 2 more variables: line <int>, vehicle <int>
```

The `visdat` package (Tierney, 2017) is also useful here, particularly to see how missing values are distributed.

```
vis_dat(delay_2019)
```



```
vis_miss(delay_2019)
```



For these known-unknowns, what we are interested in is whether they are missing at random. We want to, ideally, show that data happened to just drop out. Of course, this is unlikely the case, and so we are looking to see what is systematic about how our data are missing.

### 14.2.5 Duplicate rows

Sometime data happen to be duplicated. If we didn't notice this then our analysis would be wrong in ways that we'd not be able to consistently expect. There are a variety of ways to look for duplicated rows, but the `janitor::get_dups()` function from the `janitor` package (Firke, 2020) is especially useful.

```
janitor::get_dups(delay_2019)
#> No variable names specified - using all columns.
#> # A tibble: 158 x 11
#> date time day station code min_delay
#> <dttm> <chr> <chr> <chr> <chr> <dbl>
#> 1 2019-01-01 00:00:00 08:18 Tuesday DONLAN~ MUESA 5
#> 2 2019-01-01 00:00:00 08:18 Tuesday DONLAN~ MUESA 5
#> 3 2019-02-01 00:00:00 05:51 Friday SCARB ~ MRTO 10
#> 4 2019-02-01 00:00:00 05:51 Friday SCARB ~ MRTO 10
#> 5 2019-02-01 00:00:00 06:45 Friday MIDLAN~ MRWEA 3
#> 6 2019-02-01 00:00:00 06:45 Friday MIDLAN~ MRWEA 3
#> 7 2019-02-01 00:00:00 06:55 Friday LAWREN~ ERDO 0
```

```
#> 8 2019-02-01 00:00:00 06:55 Friday LAWREN~ ERDO 0
#> 9 2019-02-01 00:00:00 07:16 Friday MCCOWA~ MRWEA 5
#> 10 2019-02-01 00:00:00 07:16 Friday MCCOWA~ MRWEA 5
#> # ... with 148 more rows, and 5 more variables:
#> # min_gap <dbl>, bound <chr>, line <chr>, vehicle <dbl>,
#> # dupe_count <int>
```

Our delays dataset has quite a few duplicates. Again, we're interested in whether there is something systematic going on. Remembering that we're trying to quickly come to terms with dataset, one way forward is to flag this as an issue to come back to and explore later, and to just remove them for now.

```
delay_2019 <-
 delay_2019 %>%
 distinct()
```

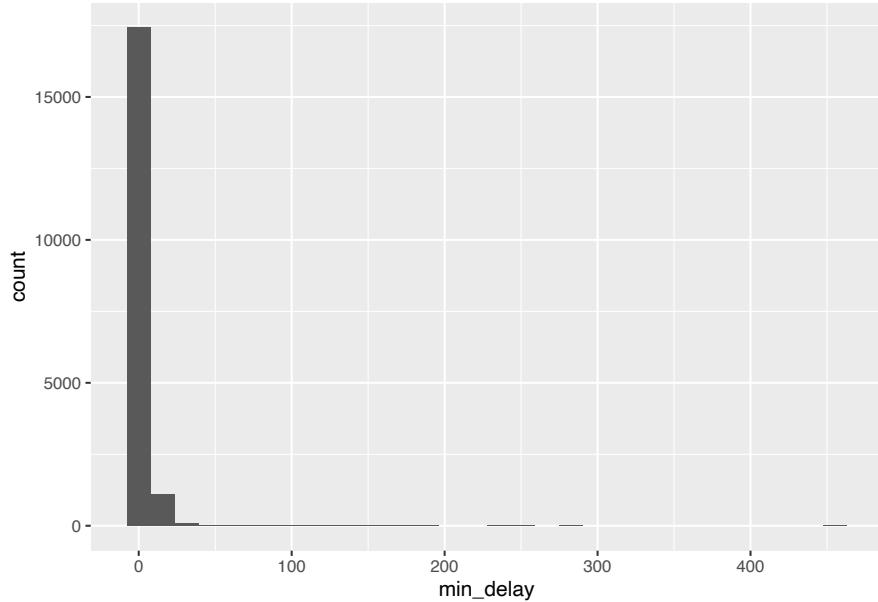
#### 14.2.6 Visualizing distributions

You need to see your data in its most raw form to understand it, and histograms, barplots, and density plots are your friends here. We're not looking for beauty here, we're looking to get a look at the data as quickly as possible.

Let's look at one outcome of interest: 'min\_delay'. First of all let's just look at a histogram of all the data.

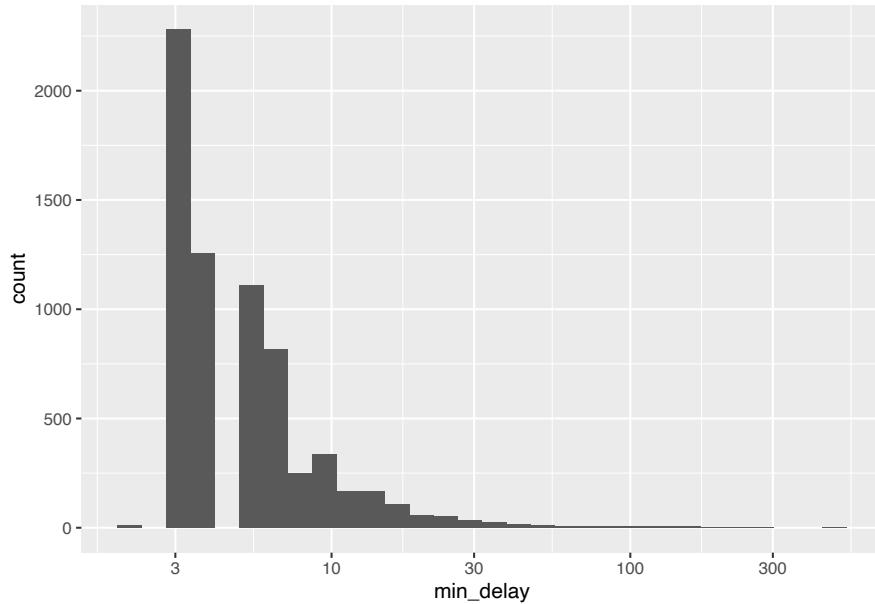
```
Removing the observations that have non-standardized lines
delay_2019 <-
 delay_2019 %>%
 filter(line %in% c("BD", "YU", "SHP", "SRT"))

ggplot(data = delay_2019) +
 geom_histogram(aes(x = min_delay))
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
```



Somewhat concerningly we have some evidence of outliers (given the large x-axis). There are a variety of ways to focus on what is going on, but a quick way is to plot it on a logged scale (remembering that we'd expect any values of 0 to drop away).

```
ggplot(data = delay_2019) +
 geom_histogram(aes(x = min_delay)) +
 scale_x_log10()
#> Warning: Transformation introduced infinite values in
#> continuous x-axis
#> `stat_bin()` using `bins = 30`. Pick better value with
#> `binwidth`.
#> Warning: Removed 11944 rows containing non-finite values
#> (stat_bin).
```



This initial exploration is further hinting at an outlying delay time, so let's take a look at the largest delays. We need to join this dataset with the 'delay\_codes' dataset to see what the delay is, and this requires some wrangling because of slightly different codes.

```
delay_2019 <-
 delay_2019 %>%
 left_join(delay_codes %>%
 rename(code = `SUB RMENU CODE`, code_desc = `CODE DESCRIPTION...3`) %>%
 select(code, code_desc)
)
#> Joining, by = "code"

delay_2019 <-
 delay_2019 %>%
 mutate(code_srt = ifelse(line=="SRT", code, "NA")) %>%
 left_join(delay_codes %>%
 rename(code_srt = `SRT RMENU CODE`, code_desc_srt = `CODE DESCRIPTION...7`) %>%
 select(code_srt, code_desc_srt)) %>%
 mutate(code = ifelse(code_srt=="NA", code, code_srt),
 code_desc = ifelse(is.na(code_desc_srt), code_desc, code_desc_srt)) %>%
 select(-code_srt, -code_desc_srt)
#> Joining, by = "code_srt"
```

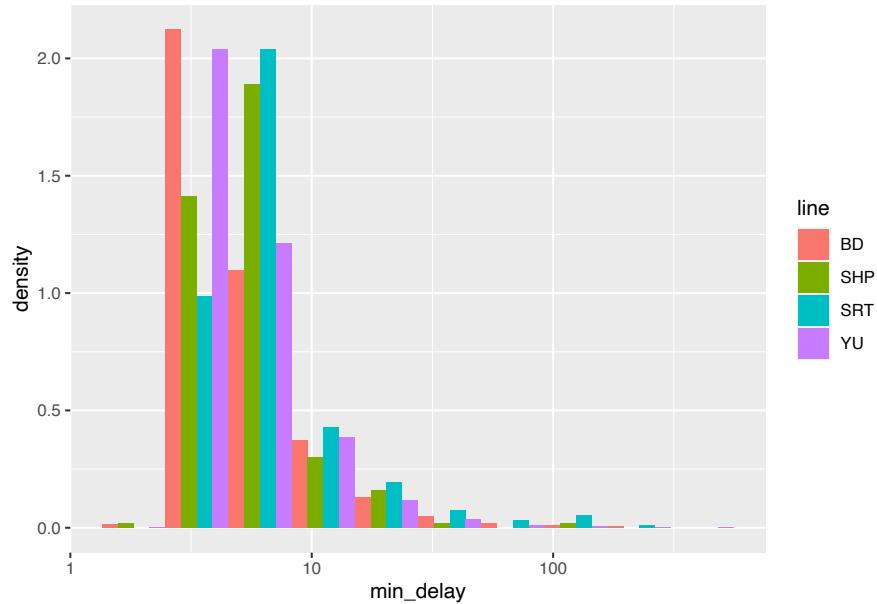
And so we can see that the 455 minute delay was due to ‘Rail Related Problem’ and seems to very much be an outlier.

```
delay_2019 %>%
 left_join(delay_codes %>%
 rename(code = `SUB RMENU CODE`, code_desc = `CODE DESCRIPTION...3`) %>%
 select(code, code_desc)) %>%
 arrange(-min_delay) %>%
 select(date, time, station, line, min_delay, code, code_desc)
#> Joining, by = c("code", "code_desc")
#> # A tibble: 18,697 x 7
#> date time station line min_delay code
#> <dttm> <chr> <chr> <dbl> <chr>
#> 1 2019-06-25 00:00:00 18:48 WILSON T~ YU 455 PPUTR
#> 2 2019-02-12 00:00:00 20:28 LAWRENCE~ SRT 284 MRWEA
#> 3 2019-06-05 00:00:00 12:42 UNION TO~ YU 250 MUPLA
#> 4 2019-10-22 00:00:00 14:22 LAWRENCE~ YU 228 PUTS
#> 5 2019-09-26 00:00:00 11:38 YORK MIL~ YU 193 MUPR1
#> 6 2019-06-08 00:00:00 08:51 SPADINA ~ BD 180 MUPLB
#> 7 2019-12-02 00:00:00 06:59 DUNDAS W~ BD 176 MUPLB
#> 8 2019-01-29 00:00:00 05:46 VICTORIA~ BD 174 MUWEA
#> 9 2019-02-22 00:00:00 17:32 ELLESMER~ SRT 168 PRW
#> 10 2019-02-10 00:00:00 07:53 BAYVIEW ~ SHP 165 PUSI
#> # ... with 18,687 more rows, and 1 more variable:
#> # code_desc <chr>
```

### 14.2.7 Groups and small counts

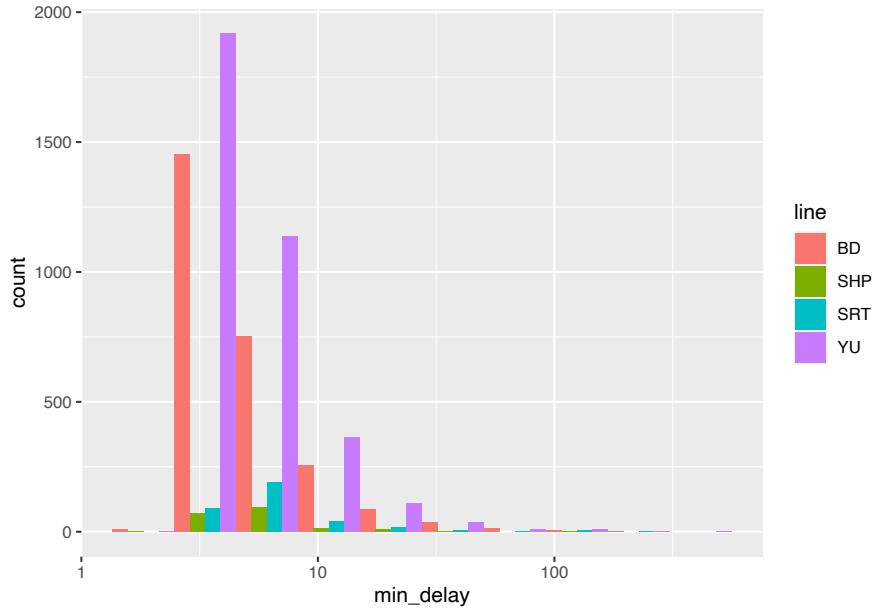
Another thing that we’re looking for is various groupings of the data, especially where sub-groups may end up with small numbers of observations in them (because any analysis could be easily influenced by them). A quick way to do this is to group the data by a variable that is of interest, for instance ‘line’, using colour.

```
ggplot(data = delay_2019) +
 geom_histogram(aes(x = min_delay, y = ..density.., fill = line),
 position = 'dodge',
 bins = 10) +
 scale_x_log10()
#> Warning: Transformation introduced infinite values in
#> continuous x-axis
#> Warning: Removed 11944 rows containing non-finite values
#> (stat_bin).
```



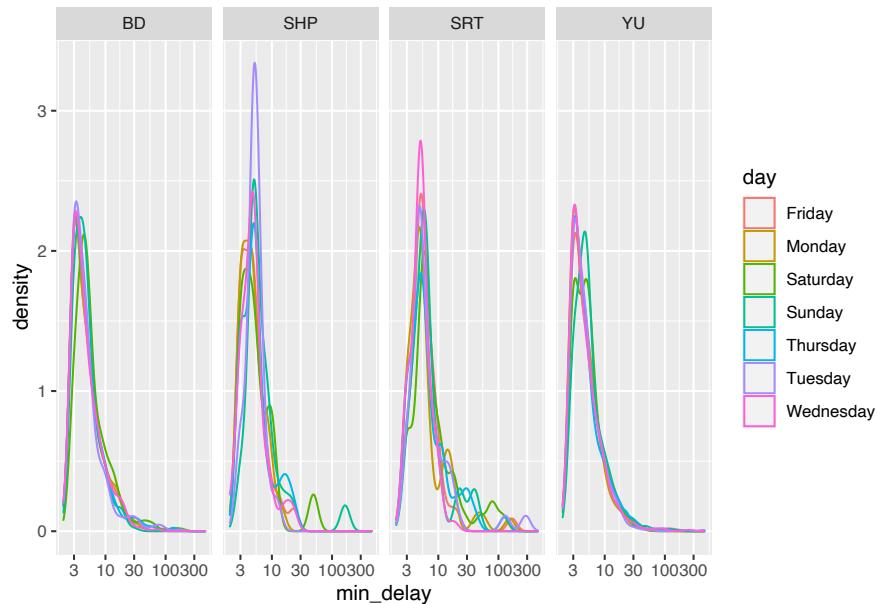
That plot uses density so that we can look at the distributions more comparably, but we should also be aware of differences in frequency. In this case, we'll see that SHP and SRT have much smaller counts.

```
ggplot(data = delay_2019) +
 geom_histogram(aes(x = min_delay, fill = line),
 position = 'dodge',
 bins = 10) +
 scale_x_log10()
#> Warning: Transformation introduced infinite values in
#> continuous x-axis
#> Warning: Removed 11944 rows containing non-finite values
#> (stat_bin).
```



To group by a second variable it can be useful to use facets. They're a little fiddly initially, but once you get used to them, they're both quick and powerful.

```
ggplot(data = delay_2019) +
 geom_density(aes(x = min_delay, color = day),
 bw = .08) +
 scale_x_log10() +
 facet_grid(~line)
#> Warning: Transformation introduced infinite values in
#> continuous x-axis
#> Warning: Removed 11944 rows containing non-finite values
#> (stat_density).
```

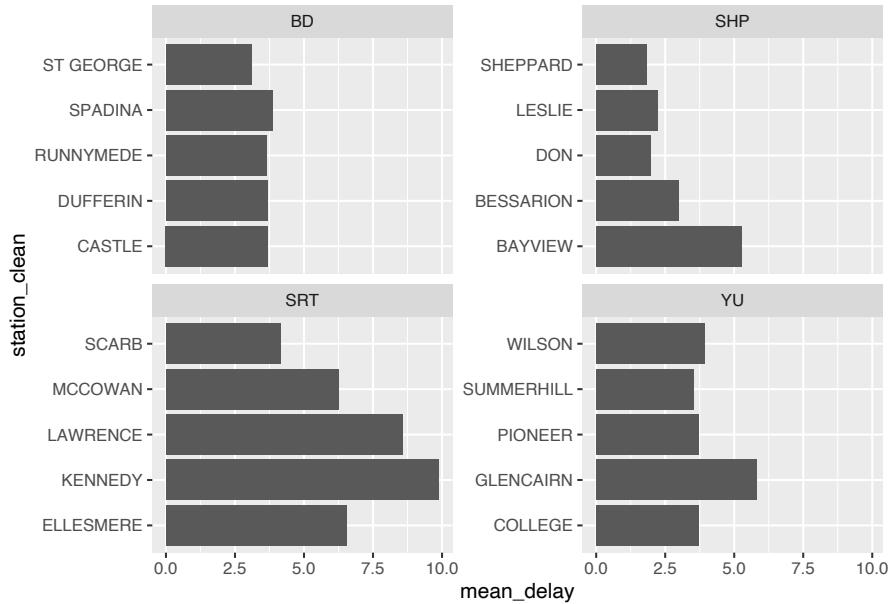


As an aside, the station names are a mess. We could try to quickly bring a little order to the chaos by just taking just the first word (or, the first two if it starts with ‘ST’).

```
delay_2019 <-
 delay_2019 %>%
 mutate(station_clean = ifelse(str_starts(station, "ST"), word(station, 1,2), word(station, 1)))
```

We can now plot the top five stations by mean delay.

```
delay_2019 %>%
 group_by(line, station_clean) %>%
 summarise(mean_delay = mean(min_delay), n_obs = n()) %>%
 filter(n_obs>1) %>%
 arrange(line, -mean_delay) %>%
 slice(1:5) %>%
 ggplot(aes(station_clean, mean_delay)) +
 geom_col() +
 coord_flip() +
 facet_wrap(~line, scales = "free_y")
#> `summarise()` has grouped output by 'line'. You can override using the `groups` argument.
```

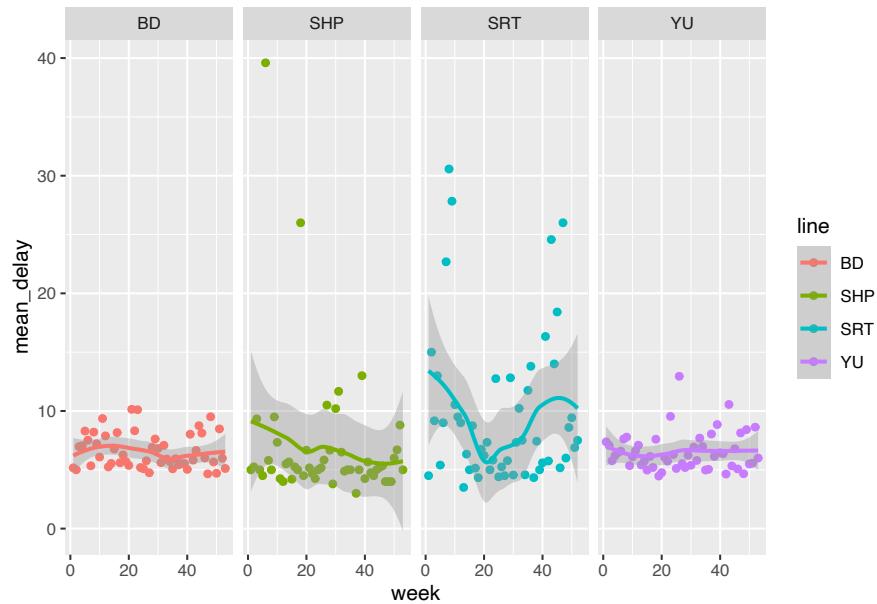


### 14.2.8 Visualizing time series

Dates are a pain to work with. They always go wrong and give issues, and so they are a critical aspect to explore during EDA. The daily plot of this data is, well, very messy (you can check for yourself). Instead, let's look by week to see if there's any seasonality. The `lubridate` package ([Grolemund and Wickham, 2011](#)) has a lot of helpful functions that deal with date variables. It's essentially indispensable.

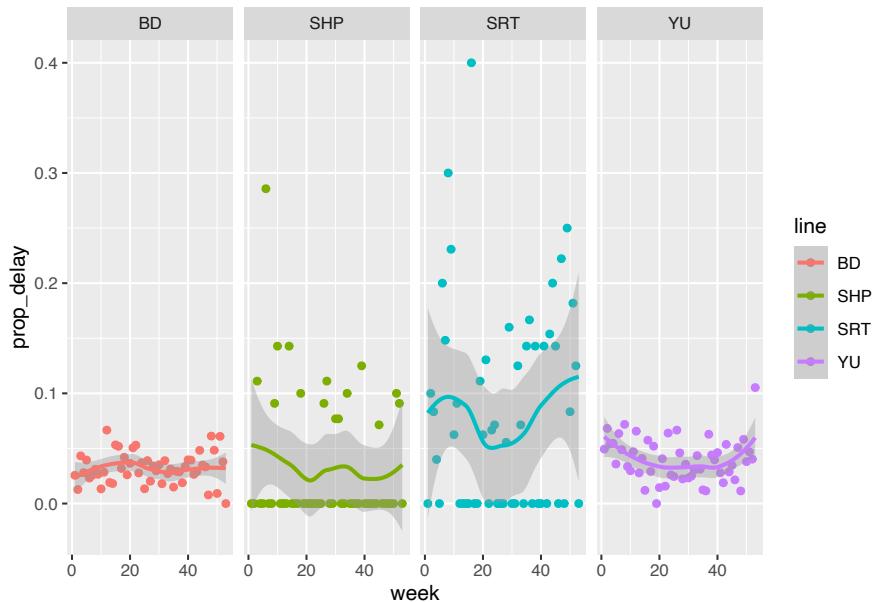
To get started, let's look at mean delay (of those that were delayed more than zero minutes).

```
delay_2019 %>%
 filter(min_delay>0) %>%
 mutate(week = week(date)) %>%
 group_by(week, line) %>%
 summarise(mean_delay = mean(min_delay)) %>%
 ggplot(aes(week, mean_delay, color = line)) +
 geom_point() +
 geom_smooth() +
 facet_grid(~line)
#> `summarise()` has grouped output by 'week'. You can override using the `groups` argument.
#> `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Now let's look at the proportion of delays that were greater than 10 minutes.

```
delay_2019 %>%
 mutate(week = week(date)) %>%
 group_by(week, line) %>%
 summarise(prop_delay = sum(min_delay>10)/n()) %>%
 ggplot(aes(week, prop_delay, color = line)) +
 geom_point() +
 geom_smooth() +
 facet_grid(~line)
#> `summarise()` has grouped output by 'week'. You can override using the `groups` argument.
#> `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

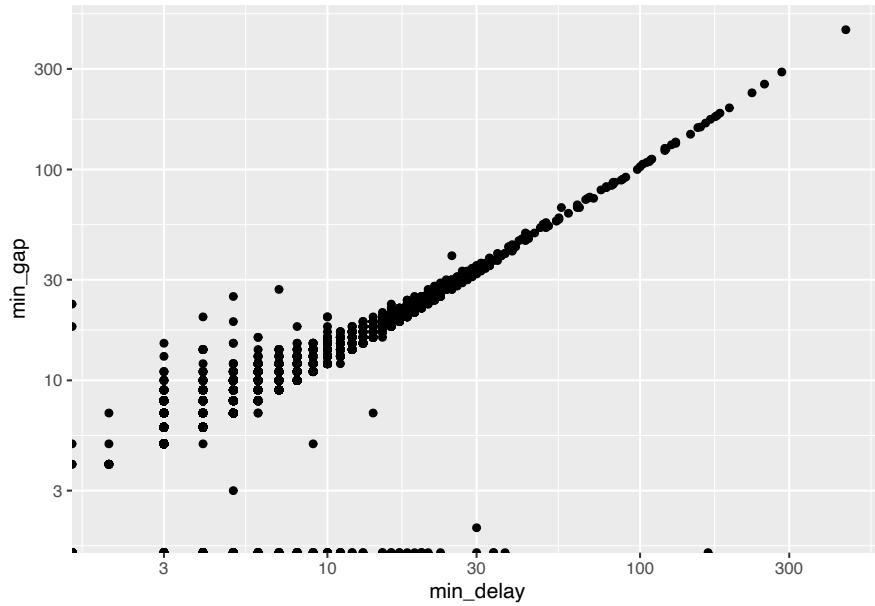


Again, and it's important to be clear here. These charts and tables and analyse have no place in a final report, but what they are doing is allowing you to become comfortable with the data. If you were doing this yourself, you should additionally be making notes about each plot and table as you go, noting the warnings and any implications or aspects to return to.

#### 14.2.9 Visualizing relationships

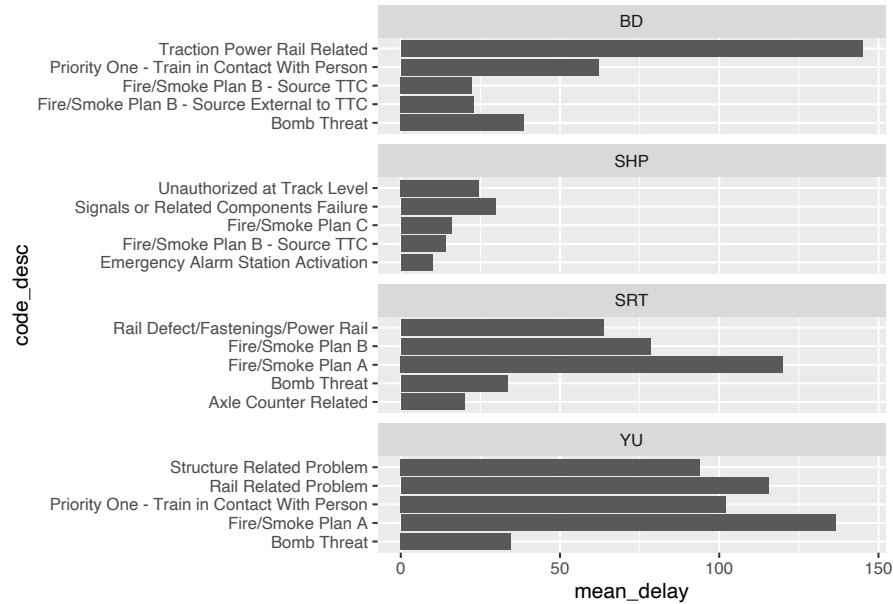
We are also interested in looking at the relationship between two variables. Scatter plots are especially useful here for continuous variables, and are a good precursor to modeling. We saw a little of that with ‘mean\_delay’ and ‘week’.

```
delay_2019 %>%
 ggplot(aes(x = min_delay, y = min_gap)) +
 geom_point() +
 scale_x_log10() +
 scale_y_log10()
#> Warning: Transformation introduced infinite values in
#> continuous x-axis
#> Warning: Transformation introduced infinite values in
#> continuous y-axis
```



The relationship between categorical variables takes more work, but we could also, for instance, look at the top five reasons for delay by station. In particular, we may be interested in whether they differ, and how any difference could be modelled.

```
delay_2019 %>%
 group_by(line, code_desc) %>%
 summarise(mean_delay = mean(min_delay)) %>%
 arrange(-mean_delay) %>%
 slice(1:5) %>%
 ggplot(aes(x = code_desc,
 y = mean_delay)) +
 geom_col() +
 facet_wrap(vars(line),
 scales = "free_y",
 nrow = 4) +
 coord_flip()
#> `summarise()` has grouped output by 'line'. You can override using the `groups` argument.
```



#### 14.2.10 Principal components analysis

Principal components analysis (PCA) is another powerful exploratory tool. It allows you to pick up potential clusters and/or outliers that can help to inform model building. To see this let's do a quick (and imperfect) example looking at types of delays by station.

The delay categories are a bit of a mess, and there's hundreds of them. As a simple start, remembering that our task is to come to terms with the dataset as quickly as possible, let's just take the first word.

```
delay_2019 <- delay_2019 %>%
 mutate(code_red = case_when(
 str_starts(code_desc, "No") ~ word(code_desc, 1, 2),
 str_starts(code_desc, "Operator") ~ word(code_desc, 1,2),
 TRUE ~ word(code_desc,1))
)
```

Let's also just restrict the analysis to causes that happen at least 50 times over 2019. To do the PCA, the dataframe also needs to be switched to wide format.

```
dwide <- delay_2019 %>%
```

```

group_by(line, station_clean) %>%
 mutate(n_obs = n()) %>%
 filter(n_obs > 1) %>%
 group_by(code_red) %>%
 mutate(tot_delay = n()) %>%
 arrange(tot_delay) %>%
 filter(tot_delay > 50) %>%
 group_by(line, station_clean, code_red) %>%
 summarise(n_delay = n()) %>%
 pivot_wider(names_from = code_red, values_from = n_delay) %>%
 mutate_all(.funs = funs(ifelse(is.na(.), 0, .)))
#> `summarise()` has grouped output by 'line', 'station_clean'. You can override using the `$.group` argument.
#> `mutate_all()` ignored the following grouping variables:
#> Columns `line`, `station_clean`
#> Use `mutate_at(df, vars(-group_cols()), myoperation)` to silence the message.
#> Warning: `funns()` was deprecated in dplyr 0.8.0.
#> Please use a list of either functions or lambdas:
#>
#> # Simple named list:
#> list(mean = mean, median = median)
#>
#> # Auto named with `tibble::lst()`:
#> tibble::lst(mean, median)
#>
#> # Using lambdas
#> list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
#> This warning is displayed once every 8 hours.
#> Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.

```

Now we can quickly do some PCA.

```

delay_pca <- prcomp(dwide[, 3:ncol(dwide)])

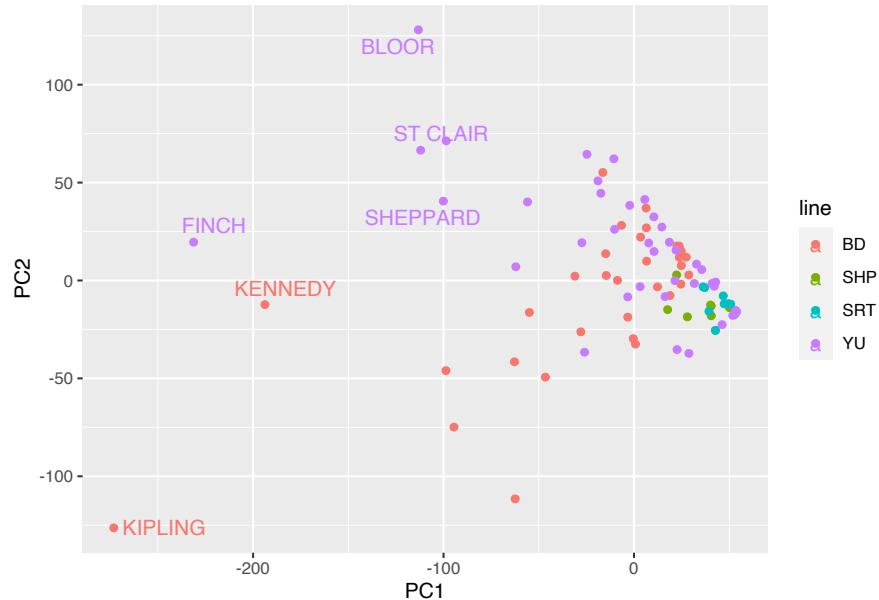
df_out <- as_tibble(delay_pca$x)
df_out <- bind_cols(dwide %>% select(line, station_clean), df_out)
head(df_out)
#> # A tibble: 6 x 40
#> # Groups: line, station_clean [6]
#> line station_clean PC1 PC2 PC3 PC4 PC5
#> <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 BD BATHURST 6.50 26.9 -2.71 -10.8 -8.40
#> 2 BD BAY 24.8 7.63 -2.19 -7.05 0.714
#> 3 BD BLOOR -62.4 -112. 57.3 -23.4 -5.09

```

```
#> 4 BD BROADVIEW -6.60 28.1 -1.06 -14.0 -6.49
#> 5 BD CASTLE 23.8 11.8 -1.31 -7.93 -3.62
#> 6 BD CHESTER 24.6 -1.87 -18.6 2.75 1.85
#> # ... with 33 more variables: PC6 <dbl>, PC7 <dbl>,
#> # PC8 <dbl>, PC9 <dbl>, PC10 <dbl>, PC11 <dbl>,
#> # PC12 <dbl>, PC13 <dbl>, PC14 <dbl>, PC15 <dbl>,
#> # PC16 <dbl>, PC17 <dbl>, PC18 <dbl>, PC19 <dbl>,
#> # PC20 <dbl>, PC21 <dbl>, PC22 <dbl>, PC23 <dbl>,
#> # PC24 <dbl>, PC25 <dbl>, PC26 <dbl>, PC27 <dbl>,
#> # PC28 <dbl>, PC29 <dbl>, PC30 <dbl>, PC31 <dbl>, ...
```

We can plot the first two principal components, and add labels to some of the outlying stations.

```
ggplot(df_out,aes(x=PC1,y=PC2,color=line)) +
 geom_point() +
 geom_text_repel(data = df_out %>% filter(PC2>100|PC1<100*-1),
 aes(label = station_clean)
)
```

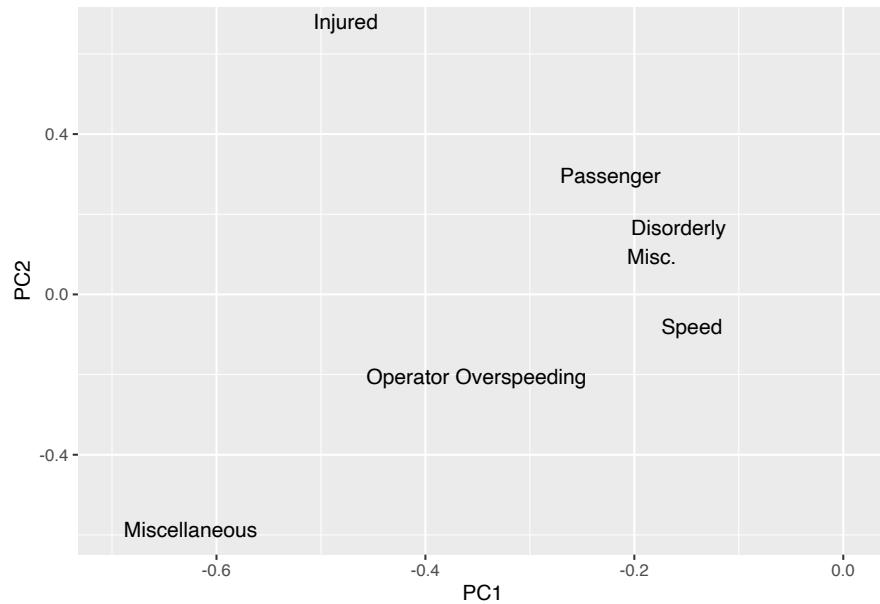


We could also plot the factor loadings. We see some evidence that perhaps one is to do with the public, compared with another to do with the operator.

```
df_out_r <- as_tibble(delay_pca$rotation)
df_out_r$feature <- colnames(dwide[,3:ncol(dwide)])

df_out_r
#> # A tibble: 38 x 39
#> PC1 PC2 PC3 PC4 PC5 PC6
#> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 -0.0412 0.0638 0.0133 -0.0467 0.0246 0.0184
#> 2 -0.0332 -0.00469 -0.0414 -0.00751 0.0201 -0.0122
#> 3 -0.135 0.207 0.0237 -0.144 0.135 -0.0381
#> 4 -0.0652 0.0475 -0.0443 -0.0251 -0.00139 -0.0748
#> 5 -0.00443 0.00878 -0.0000499 -0.000830 0.00967 0.00954
#> 6 -0.0268 -0.00722 -0.00439 0.000534 -0.0151 -0.0125
#> 7 -0.0813 0.0960 -0.0462 0.0479 -0.0978 -0.0365
#> 8 -0.0117 0.0135 0.00548 -0.0294 0.0125 0.0377
#> 9 -0.516 0.655 -0.0177 -0.162 -0.221 -0.287
#> 10 -0.151 0.0826 0.0548 0.352 -0.397 0.281
#> # ... with 28 more rows, and 33 more variables: PC7 <dbl>,
#> # PC8 <dbl>, PC9 <dbl>, PC10 <dbl>, PC11 <dbl>,
#> # PC12 <dbl>, PC13 <dbl>, PC14 <dbl>, PC15 <dbl>,
#> # PC16 <dbl>, PC17 <dbl>, PC18 <dbl>, PC19 <dbl>,
#> # PC20 <dbl>, PC21 <dbl>, PC22 <dbl>, PC23 <dbl>,
#> # PC24 <dbl>, PC25 <dbl>, PC26 <dbl>, PC27 <dbl>,
#> # PC28 <dbl>, PC29 <dbl>, PC30 <dbl>, PC31 <dbl>, ...

ggplot(df_out_r,aes(x=PC1,y=PC2,label=feature)) + geom_text_repel()
#> Warning: ggrepel: 31 unlabeled data points (too many
#> overlaps). Consider increasing max.overlaps
```



## 14.3 Case study - Opinions about a casino in Toronto

This was written by Michael Chong<sup>3</sup>.

### 14.3.1 Data preparation

Here we use the `opendatatoronto` package again. See the previous case study for a deeper explanation of how the code below works.

The dataset I'm extracting below are the results from a survey in 2012 regarding the establishment of a casino in Toronto. More info available by following this link<sup>4</sup>. In this analysis, we'll be hoping to address the question: which demographic (age/gender) groups are more likely to be supportive of a new casino in Toronto?

```
Get the data
casino_resource <-
 search_packages("casino_survey")%>%
 list_package_resources() %>%
```

<sup>3</sup><https://michael-chong.github.io>

<sup>4</sup><https://open.toronto.ca/dataset/casino-survey-results/>

```

filter(name == "toronto-casino-survey-results") %>%
get_resource()
#> New names:
#> * ` ` -> ...93
#> * ` ` -> ...94
head(casino_resource)
#> $tblSurvey
#> # A tibble: 17,766 x 94
#> SurveyID Q1_A Q1_B1 Q1_B2 Q1_B3 Q2_A Q2_B Q3_A Q3_B
#> <dbl> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 1 Stron~ Do n~ Do n~ Do n~ Does~ "As ~ Not ~ Very~
#> 2 2 Stron~ Econ~ Jobs Arts~ Fits~ "Cos~ Very~ Very~
#> 3 3 Stron~ Ther~ If t~ <NA> Fits~ "Big~ Very~ Very~
#> 4 4 Somew~ beli~ mone~ evid~ Does~ "My ~ Very~ Very~
#> 5 5 Neutr~ Like~ Conc~ <NA> Neut~ "Aga~ Very~ Very~
#> 6 6 Stron~ have~ <NA> <NA> Does~ "Tor~ Not ~ Not ~
#> 7 7 Stron~ The ~ Peop~ We s~ Does~ "#3 ~ Not ~ Not ~
#> 8 8 Stron~ It w~ Mora~ <NA> Does~ "Cas~ Very~ Very~
#> 9 9 Stron~ It's~ traf~ heal~ Does~ "No ~ Not ~ Very~
#> 10 10 Stron~ Toro~ Avoi~ Prov~ Fits~ "Tor~ Very~ Very~
#> # ... with 17,756 more rows, and 85 more variables:
#> # Q3_C <chr>, Q3_D <chr>, Q3_E <chr>, Q3_F <chr>,
#> # Q3_G <chr>, Q3_H <chr>, Q3_I <chr>, Q3_J <chr>,
#> # Q3_K <chr>, Q3_L <chr>, Q3_M <chr>, Q3_N <chr>,
#> # Q3_O <chr>, Q3_P <chr>, Q3_Q <chr>, Q3_Q_Other <chr>,
#> # Q3_Comments <chr>, Q4_A <chr>, Q5 <chr>, Q6 <chr>,
#> # Q6_Comments <chr>, Q7_A_StandAlone <chr>, ...
#>
#> $Sheet1
#> # A tibble: 0 x 0

```

The object `casino_resource` isn't quite useable yet, because it's (inconveniently) stored as a list of 2 data frames:

```

Check what kind of object the casino_resource object is
class(casino_resource)
#> [1] "list"

```

If we just return the object, we can see that the second list item is empty, and we just want to keep the first one:

```

casino_resource
#> $tblSurvey

```

```
#> # A tibble: 17,766 x 94
#> SurveyID Q1_A Q1_B1 Q1_B2 Q1_B3 Q2_A Q2_B Q3_A Q3_B
#> <dbl> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 1 Stron~ Do n~ Do n~ Do n~ Does~ "As ~ Not ~ Very~
#> 2 2 Stron~ Econ~ Jobs Arts~ Fits~ "Cos~ Very~ Very~
#> 3 3 Stron~ Ther~ If t~ <NA> Fits~ "Big~ Very~ Very~
#> 4 4 Somew~ belie~ mone~ evid~ Does~ "My ~ Very~ Very~
#> 5 5 Neutr~ Like~ Conc~ <NA> Neut~ "Aga~ Very~ Very~
#> 6 6 Stron~ have~ <NA> <NA> Does~ "Tor~ Not ~ Not ~
#> 7 7 Stron~ The ~ Peop~ We s~ Does~ "#3 ~ Not ~ Not ~
#> 8 8 Stron~ It w~ Mora~ <NA> Does~ "Cas~ Very~ Very~
#> 9 9 Stron~ It's~ traf~ heal~ Does~ "No ~ Not ~ Very~
#> 10 10 Stron~ Toro~ Avoi~ Prov~ Fits~ "Tor~ Very~ Very~
#> # ... with 17,756 more rows, and 85 more variables:
#> # Q3_C <chr>, Q3_D <chr>, Q3_E <chr>, Q3_F <chr>,
#> # Q3_G <chr>, Q3_H <chr>, Q3_I <chr>, Q3_J <chr>,
#> # Q3_K <chr>, Q3_L <chr>, Q3_M <chr>, Q3_N <chr>,
#> # Q3_O <chr>, Q3_P <chr>, Q3_Q <chr>, Q3_Q_Other <chr>,
#> # Q3_Comments <chr>, Q4_A <chr>, Q5 <chr>, Q6 <chr>,
#> # Q6_Comments <chr>, Q7_A_StandAlone <chr>, ...
#>
#> $Sheet1
#> # A tibble: 0 x 0
```

So, let's only keep the first item by indexing the list with double square brackets:

```
casino_data <- casino_resource[[1]]
```

Let's check out what the first couple rows of the dataframe looks like. By default, `head()` returns the first 6 rows:

```
head(casino_data)
#> # A tibble: 6 x 94
#> SurveyID Q1_A Q1_B1 Q1_B2 Q1_B3 Q2_A Q2_B Q3_A Q3_B
#> <dbl> <chr> <chr> <chr> <chr> <chr> <chr> <chr>
#> 1 1 Stron~ Do no~ Do n~ Do n~ Does~ "As ~ Not ~ Very~
#> 2 2 Stron~ Econo~ Jobs Arts~ Fits~ "Cos~ Very~ Very~
#> 3 3 Stron~ There~ If t~ <NA> Fits~ "Big~ Very~ Very~
#> 4 4 Somew~ belie~ mone~ evid~ Does~ "My ~ Very~ Very~
#> 5 5 Neutr~ Like~ Conc~ <NA> Neut~ "Aga~ Very~ Very~
#> 6 6 Stron~ have~ <NA> <NA> Does~ "Tor~ Not ~ Not ~
#> # ... with 85 more variables: Q3_C <chr>, Q3_D <chr>,
```

```
#> # Q3_E <chr>, Q3_F <chr>, Q3_G <chr>, Q3_H <chr>,
#> # Q3_I <chr>, Q3_J <chr>, Q3_K <chr>, Q3_L <chr>,
#> # Q3_M <chr>, Q3_N <chr>, Q3_O <chr>, Q3_P <chr>,
#> # Q3_Q <chr>, Q3_Q_Other <chr>, Q3_Comments <chr>,
#> # Q4_A <chr>, Q5 <chr>, Q6 <chr>, Q6_Comments <chr>,
#> # Q7_A_StandAlone <chr>, Q7_A_Integrated <chr>, ...
```

Unfortunately the column names aren't very informative. For simplicity, we'll use the 'pdf' questionnaire that accompanies this dataset from the Toronto Open Data website. Alternatively, we could get and parse the 'readme' through the R package. Here's a link to the questionnaire<sup>5</sup>.

Question 1 indicates the level of support for a casino in Toronto. We'll use this as the response variable.

Concerning potential predictor variables, most of the questions ask respondents about their opinions on different aspects of a potential casino development, which aren't particularly useful towards our cause. The only demographic variables are `Age` and `Gender`, so let's choose these.

Here I'm also going to rename the columns so that my resulting data frame has columns 'opinion', 'age', and 'gender'.

```
Narrow down the dataframe to our variables of interest
casino_data <-
 casino_data %>%
 select(Q1_A, Age, Gender) %>%
 rename(opinion = Q1_A, age = Age, gender = Gender)

Look at first couple rows:
head(casino_data)
#> # A tibble: 6 x 3
#> opinion age gender
#> <chr> <chr> <chr>
#> 1 Strongly Opposed 25-34 Male
#> 2 Strongly in Favour 35-44 Female
#> 3 Strongly in Favour 55-64 Male
#> 4 Somewhat Opposed 25-34 Male
#> 5 Neutral or Mixed Feelings 25-34 Female
#> 6 Strongly Opposed 45-54 Female
```

---

<sup>5</sup><https://ckan0.cf.opendata.inter.prod-toronto.ca/dataset/427ca4cd-168a-4a37-883d-4a574277caf5/resource/ae135d6a-6921-4905-bc79-516fcfd428b7b/download/toronto-casino-survey-feedback-form.pdf>

### 14.3.2 Some visual exploration (and more cleanup, of course)

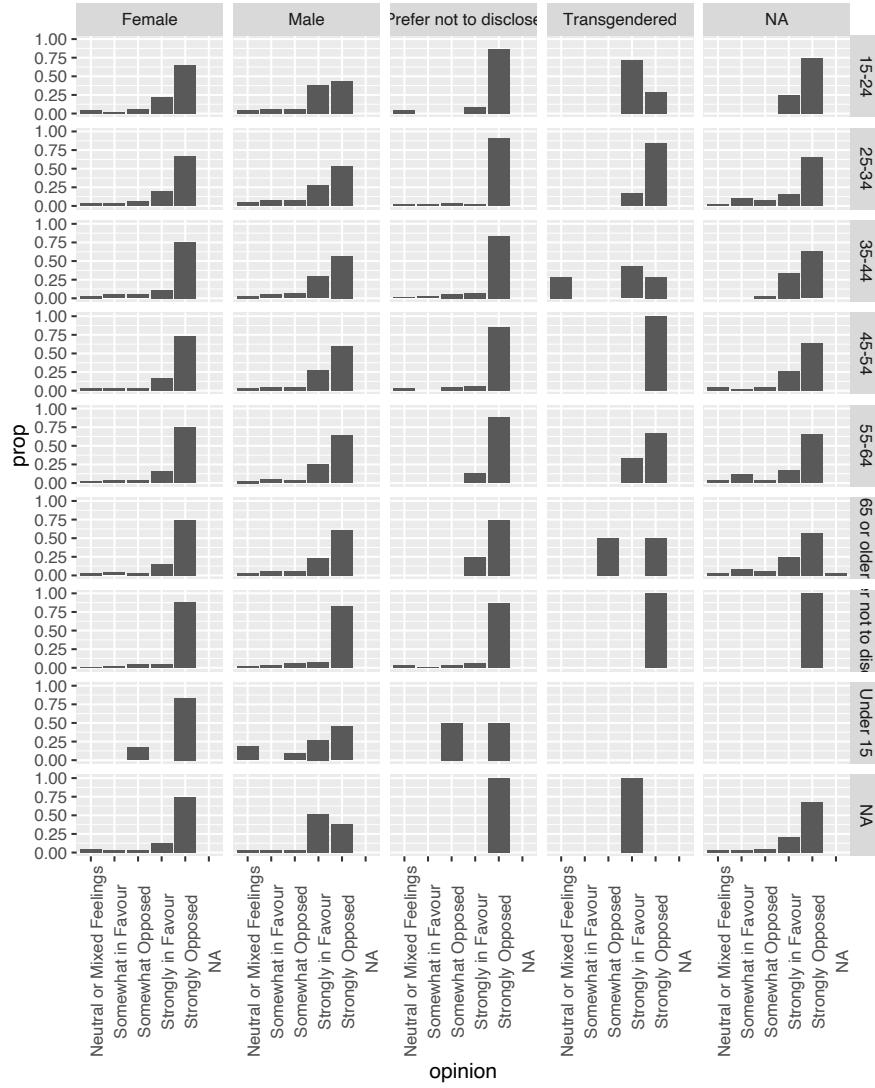
Let's first do some quick exploration to get a feel for what's going on in the data. We'll first calculate proportions of casino support for each age-gender combination:

```
Calculate proportions
casino_summary <- casino_data %>%
 group_by(age, gender, opinion) %>%
 summarise(n = n()) %>% # Count the number in each group and response
 group_by(age, gender) %>%
 mutate(prop = n/sum(n)) # Calculate proportions within each group
#> `summarise()` has grouped output by 'age', 'gender'. You can override using the `groups` argument
```

Some notes:

- we use `geom_col()` to make a bar chart,
- `facet_grid()` modifies the plot so that the plot has panels that correspond only to certain values of discrete variables (in this case, we will “facet” by age and gender). This is helpful in this case because we are interested in how the distribution of opinions changes by age and gender.

```
ggplot(casino_summary) +
 geom_col(aes(x = opinion, y = prop)) + # Specify a histogram of opinion responses
 facet_grid(age~gender) + #Facet by age and gender
 theme(axis.text.x = element_text(angle = 90)) # Rotate the x-axis labels to be readable
```



Some things to note:

- the x-axis labels are out of order in the sense that they are not in a monotone order of increasing/decreasing support
- there are NA values in opinion, age, and gender, as well as “Prefer not to disclose” responses

### 14.3.3 Getting the data into a more model-suitable format

First we need to get rid of responses that aren't suitable. For simplicity we'll assume that `NA` values and "Prefer not to disclose" responses occur randomly, and remove them from our dataset (note in reality this assumption might not hold up and we might want to be more careful). Let's check how many rows are in the original dataset:

```
nrow() returns the number of rows in a dataframe:
nrow(casino_data)
#> [1] 17766
```

Now let's `dplyr::filter()` accordingly to omit the responses we don't want. In case you're unfamiliar, I'm going to make use of:

- `is.na()`, which returns `TRUE` if the argument is `NA`,
- the `!` operator, which flips `TRUE` and `FALSE`. So for instance, `!is.na(x)` will return `TRUE` if `x` is NOT `NA`, which is what we want to keep.

```
casino_data <- casino_data %>%
 # Only keep rows with non-NA:
 filter(!is.na(opinion), !is.na(age), !is.na(gender)) %>%
 # Only keep rows where age and gender are disclosed:
 filter(age != "Prefer not to disclose", gender != "Prefer not to disclose")
```

Let's check how many rows of data we're left with:

```
nrow(casino_data)
#> [1] 13658
```

Now we need to convert the response variable into binary. To clean up the first problem (response variables out of order), we might as well take this opportunity to convert these into a format suitable for our model. In a logistic regression, we would like our response variable to be binary, but in this case we have 5 possible categories ranging from "Strongly Opposed" to "Strongly in Favour". We'll recategorize them into a new 'supportive\_or\_not' variable as follows.

- 'supportive = 1' if "Strongly in Favour" or "Somewhat in Favour"
- 'supportive = 0' if "Neutral or Mixed Feelings", "Somewhat Opposed", or "Strongly Opposed"

We do this with the `dplyr::mutate()` function, which creates new columns (possibly as functions of existing columns), and `dplyr::case_when()`, which provides a way to assign values conditional on if-statements. The syntax here is a little strange. On the LHS of the `~` is the "if" condition, and the RHS

of the tilde is the value to return. For example, ‘`x == 0 ~ 3`’ would return 3 when ‘`x`’ is 0.

Another commonly used operator here is the `%in%` operator, which checks whether something is an element of a vector, for instance, e.g.:

- 1 `%in%` `c(1, 3, 4)` returns TRUE
- 2 `%in%` `c(1, 3, 4)` returns FALSE

```
Store possible opinions in vectors
yes_opinions <- c("Strongly in Favour", "Somewhat in Favour")
no_opinions <- c("Neutral or Mixed Feelings", "Somewhat Opposed", "Strongly Opposed")

Create `supportive` column:
casino_data <-
 casino_data %>%
 mutate(supportive = case_when(
 opinion %in% yes_opinions ~ TRUE, # Assign TRUE
 opinion %in% no_opinions ~ FALSE # Assign FALSE
))
```

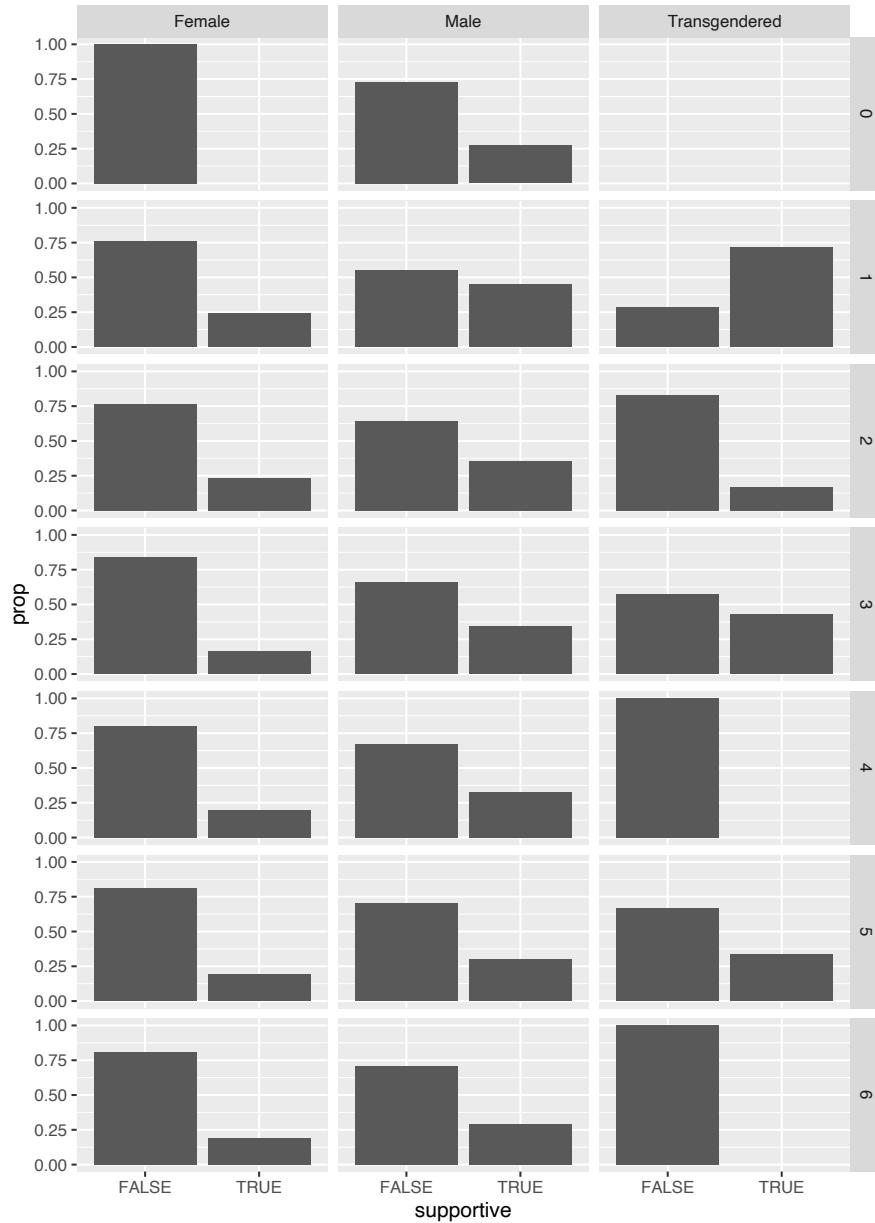
Now we need to convert age to a numeric variable. Age in this survey is given in age groups. Let’s instead map it to a numeric variable so that we can more easily talk about trends with age. We’ll map the youngest age to 1, and so on:

```
casino_data <-
 casino_data %>%
 mutate(age_group = case_when(
 age == "Under 15" ~ 0,
 age == "15-24" ~ 1,
 age == "25-34" ~ 2,
 age == "35-44" ~ 3,
 age == "45-54" ~ 4,
 age == "55-64" ~ 5,
 age == "65 or older" ~ 6
))
```

Now let’s make the same plot again, with our new processed data:

```
casino_summary2 <-
 casino_data %>%
 group_by(age_group, gender, supportive) %>%
 summarise(n = n()) %>% # Count the number in each group and response
 group_by(age_group, gender) %>%
 mutate(prop = n/sum(n)) # Calculate proportions within each group
```

```
#> `summarise()` has grouped output by 'age_group', 'gender'. You can override using the `groups`
ggplot(casino_summary2) +
 facet_grid(age_group ~ gender) +
 geom_col(aes(x = supportive, y = prop))
```



We can sort of see some difference in the distribution between different panels.  
To formalize this, we can run a logistic regression.

#### 14.3.4 Logistic Regression

Now, we're set up to feed it to the regression. We can do this with `glm()`, which allows us to fit generalized linear models.

We use `family = "binomial"` to specify a logistic regression, and our formula is `supportive ~ age_group + gender`, which indicates that `supportive` is the (binary) response variable since it's on the LHS, and `age_group` and `gender` are our predictor variables.

```
casino_glm <- glm(supportive ~ age_group + gender, data = casino_data, family = "binomial")
```

We can take a look at the results of running the GLM using `summary()`:

```
summary(casino_glm)
#>
#> Call:
#> glm(formula = supportive ~ age_group + gender, family = "binomial",
#> data = casino_data)
#>
#> Deviance Residuals:
#> Min 1Q Median 3Q Max
#> -1.0107 -0.8888 -0.6804 1.4249 1.8822
#>
#> Coefficients:
#> Estimate Std. Error z value Pr(>|z|)
#> (Intercept) -1.10594 0.05863 -18.862 < 2e-16
#> age_group -0.07983 0.01376 -5.801 6.59e-09
#> genderMale 0.70036 0.04027 17.390 < 2e-16
#> genderTransgendered 0.69023 0.39276 1.757 0.0789
#>
#> (Intercept) ***
#> age_group ***
#> genderMale ***
#> genderTransgendered .
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#> Null deviance: 16010 on 13657 degrees of freedom
#> Residual deviance: 15653 on 13654 degrees of freedom
#> AIC: 15661
```

```
#>
#> Number of Fisher Scoring iterations: 4
```

Interpretation can be a little tricky. Here are some important things to note about our results:

Firstly, we have a numeric age group variable. Remember that we coded `age_group` as numbers 1 to 5. Because we've used age groups instead of age, we have to be careful with how we phrase our conclusion. The coefficient estimate corresponds to the effect of moving up a unit on the age group scale (e.g. from the 25-34 age group to the 35-44 age group), rather than 1 year in age (e.g. from age 28 to 29).

Secondly, the results are in log-odds ratios. The effect estimates are on the log-odds scale. This means the effect of -0.07983 for `age_group` is interpreted as: 'for each unit increase in `age_group`, we estimate a 0.07983 decrease in the log-odds of being supportive of a casino'.

We could exponentiate the coefficient estimate to make this at least a little easier to interpret. The number we get is interpreted as a factor for the odds.

```
exp(-0.07983)
#> [1] 0.9232733
```

So our (cleaner) interpretation is: 'the odds of an individuals of the same gender being pro-casino are predicted to change by a factor of 0.9232733 for each unit increase in `age_group`'

Finally, we have a baseline category. First, note that because we have categorical variables, the gender coefficients are relative to a "baseline" category. The value of `gender` that doesn't appear in the table, `Female`, is implicitly used as our baseline gender category. Technical note: if the variable is stored as a `character` class, then `glm()` will choose the alphabetically first value to use as the baseline.

```
exp(0.70036)
#> [1] 2.014478
```

So, the interpretation of the `genderMale` coefficient is: 'the odds of a male individual supporting a casino is 2.014478 times higher than a female individual of the same `age_group`'.

We can make estimates in a variety of ways. First we'll look at it manually.

Using the formula found in James et al. (2017, 4.3.3), we can make estimates for an individual of certain characteristics. Suppose we wanted to predict the

the probability of supporting a Toronto casino for an individual who was 36 and identified as transgender. Then:

- `age_group` takes a value of 3, since they are in the age group of 35-44 coded as 3,
- `genderTransgendered` takes a value of 1

First, let's extract the coefficient estimates as a vector using `coefficients()`:

```
coefs <- coefficients(casino_glm)
coefs
#> (Intercept) age_group genderMale
#> -1.10593925 -0.07983372 0.70036199
#> genderTransgendered
#> 0.69022910
```

Since this vector is labeled, we can index it using square brackets and names. For instance:

```
coefs["age_group"]
#> age_group
#> -0.07983372
```

So first let's evaluate the exponent term  $e^{\beta_0 + \dots + \beta_p X_p}$ :

```
exp_term <- exp(coefs["(Intercept)"] + coefs["age_group"]*3 + coefs["genderTransgendered"]*1)
```

Now evaluate the expression that gives the probability of casino support:

```
The unname() command just takes off the label that it "inherited" from the coefs vector.
(don't worry about it, doesn't affect any functionality)
unname(exp_term / (1 + exp_term))
#> [1] 0.3418161
```

That works, but there are more stream-lined ways. Thankfully R comes with a convenient function to make prediction estimates from a `glm()`. We do this using the `predict()` function. First, we need to make a dataframe that has the relevant variables and values that we're interested in predicting. We'll use the same values as before:

```
prediction_df <- data.frame(age_group = 3, gender = "Transgendered")
```

The dataframe looks like this:

```
prediction_df
#> age_group gender
#> 1 3 Transgendered
```

Then we feed it into the `predict()` function, along with our `glm` object. To get the probability, we need to specify `type = "response"`.

```
predict(casino_glm, newdata = prediction_df, type = "response")
#> 1
#> 0.3418161
```

This matches the probability we got from doing this manually, yay!

## 14.4 Case study - Airbnb listing in Toronto

### 14.4.1 Essentials

In this case study we look at Airbnb listings in Toronto.

### 14.4.2 Set up

```
library(broom) # Helps with model outputs etc
library(here) # Helps with specifying path names
library(janitor) # Helps with initial data cleaning and pretty tables
library(tidymodels) # Help with modelling
library(tidyverse)
library(visdat) # Helps check missing values
```

### 14.4.3 Get data

The dataset is from Inside Airbnb ([Cox, 2021](#)). The `here` package will help ([Müller, 2017](#)).

We can give `read_csv()` a link to where the dataset is and it will download it. This helps with reproducibility because the source is clear. But, as that link could change at any time, longer-term reproducibility, as well as wanting to minimise the effect on the Inside Airbnb servers, suggest that we should also save a local copy of the data and then use that. (As the original data is not ours, we should not make that public without first getting written permission.)

```
For reproducibility
airbnb_data_reduced <- read_csv("http://data.insideairbnb.com/canada/on/toronto/2021-01-02/data/
write_csv(airbnb_data_reduced, "week_6/data/airbnb_toronto_2019-12-07.csv")

For actual work
airbnb_data <- read_csv(here::here("dont_push/airbnb_toronto_2021_january-listings.csv"), guess_max = 1000)

The guess_max option in read_csv helps us avoid having to specify the column types. Usually read_csv
will guess the column types based on the first few rows, but here we want to make sure we have enough information.
```

#### 14.4.4 Clean data

There are an enormous number of columns, so we'll just select a few.

```
names(airbnb_data) %>% length()
#> [1] 74

airbnb_data_selected <-
 airbnb_data %>%
 select(host_id,
 host_since,
 host_response_time,
 host_is_superhost,
 host_listings_count,
 host_total_listings_count,
 host_neighbourhood,
 host_listings_count,
 neighbourhood_cleansed,
 room_type,
 bathrooms,
 bedrooms,
 price,
 number_of_reviews,
 has_availability,
 review_scores_rating,
 review_scores_accuracy,
 review_scores_cleanliness,
 review_scores_checkin,
 review_scores_communication,
 review_scores_location,
 review_scores_value
)
```

We might like to have a brief look at the dataset to see if anything weird is going on. There are a bunch of ways of doing this.

A few things jump out:

1. There are character variables that should probably be numerics or dates/times: `host_response_time`, `price`, `weekly_price`, `monthly_price`, `cleaning_fee`.
2. Weekly and monthly price is missing in an overwhelming number of observations.
3. Roughly a fifth of observations are missing a review score and there seems like there is some correlation between those review-type variables.
4. There are two variants of the neighbourhood name.
5. There are NAs in `host_is_superhost`.
6. The reviews seem really skewed.
7. There is someone who has 328 properties on Airbnb.

#### 14.4.4.1 Price

First we need to convert to a numeric. This is a common problem, and you need to be a little careful that it doesn't all just convert to NAs. In our case if we just force the price data to be a numeric then it will go to NA because there are a lot of characters that R doesn't know how to convert, e.g. what is the numeric for '\$'? So we need to remove those characters first.

```
airbnb_data_selected$price %>% head()
#> [1] "$469.00" "$96.00" "$64.00" "$70.00" "$45.00"
#> [6] "$127.00"

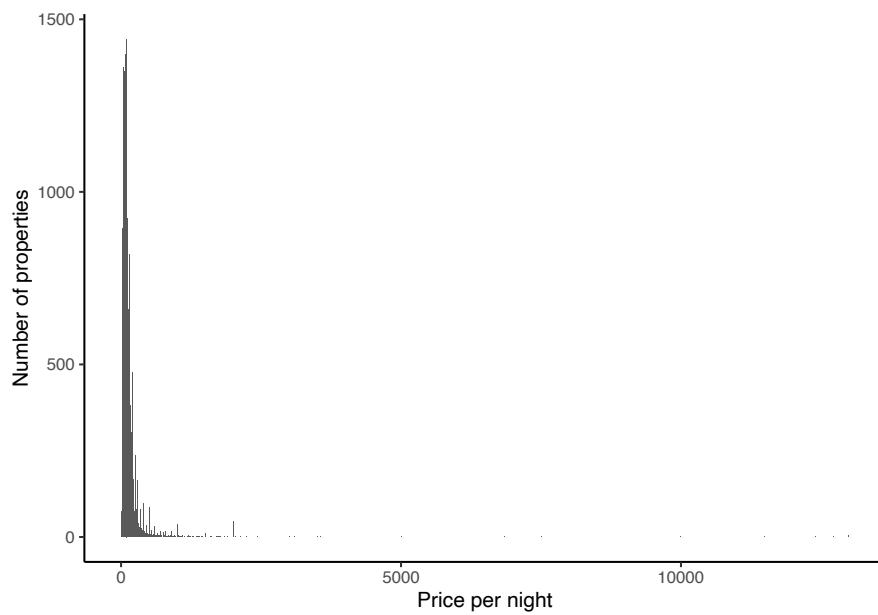
First work out what is going on
airbnb_data_selected$price %>% str_split("") %>% unlist() %>% unique()
#> [1] "$" "4" "6" "9" "." "0" "7" "5" "1" "2" "3" "8" ","
It's clear that '$' needs to go. The only odd thing is ',' so take a look at those:
airbnb_data_selected %>%
 select(price) %>%
 filter(str_detect(price, ","))

#> # A tibble: 145 x 1
#> price
#> <chr>
#> 1 $1,724.00
#> 2 $1,000.00
#> 3 $1,100.00
#> 4 $1,450.00
#> 5 $1,019.00
```

```
#> 6 $1,000.00
#> 7 $1,300.00
#> 8 $2,142.00
#> 9 $2,000.00
#> 10 $1,200.00
#> # ... with 135 more rows
It's clear that the data is just nicely formatted, but we need to remove the comma:
airbnb_data_selected <-
 airbnb_data_selected %>%
 mutate(price = str_remove(price, "\\$"),
 price = str_remove(price, ","),
 price = as.integer(price)
)
```

Now we can have a look at prices.

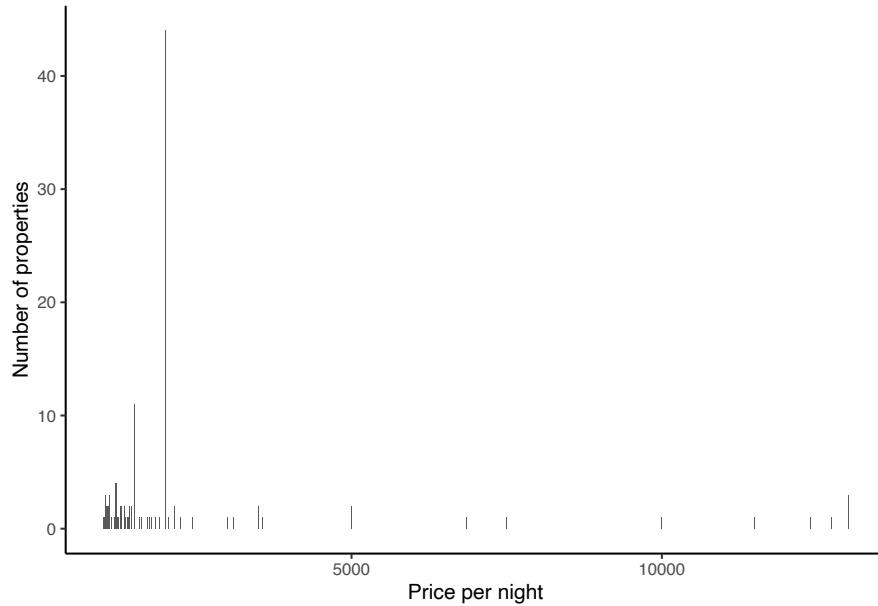
```
Look at distribution of price
airbnb_data_selected %>%
 ggplot(aes(x = price)) +
 geom_histogram(binwidth = 10) +
 theme_classic() +
 labs(x = "Price per night",
 y = "Number of properties")
```



```
We use bins with a width of 10, so that's going to aggregate prices into 10s so that we don't g
```

So we have some outliers. Let's zoom in on prices that are more than \$1,000.

```
Look at distribution of high prices
airbnb_data_selected %>%
 filter(price > 1000) %>%
 ggplot(aes(x = price)) +
 geom_histogram(binwidth = 10) +
 theme_classic() +
 labs(x = "Price per night",
 y = "Number of properties")
```



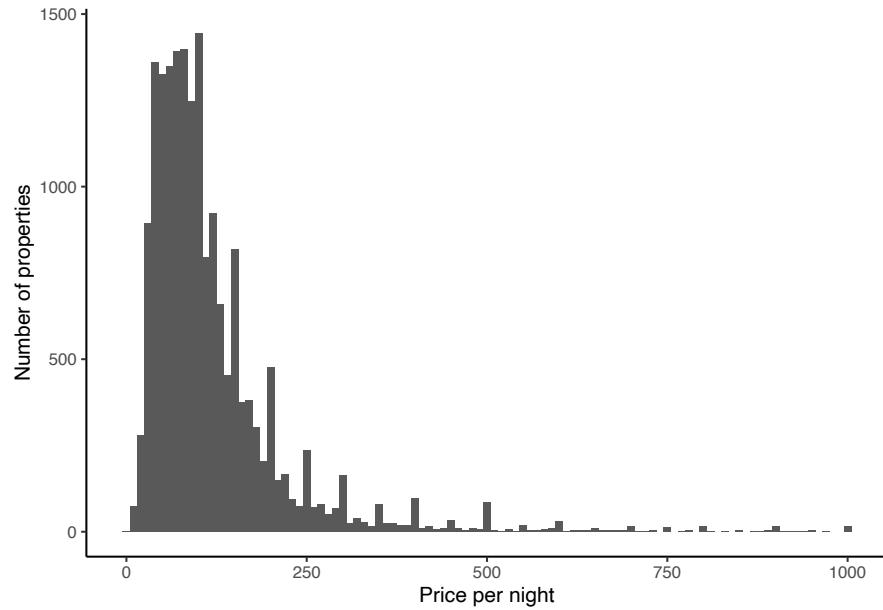
Let's look in more detail at those with a price more than \$4,999.

```
airbnb_data_selected %>%
 filter(price > 4999)
#> # A tibble: 11 x 21
#> host_id host_since host_response_time host_is_superhost
#> <dbl> <date> <chr> <lgl>
#> 1 9310264 2013-10-08 N/A FALSE
#> 2 99076885 2016-10-10 N/A FALSE
#> 3 119693302 2017-03-07 N/A FALSE
```

```
#> 4 147240941 2017-08-22 N/A FALSE
#> 5 174625477 2018-02-21 N/A FALSE
#> 6 70349386 2016-05-04 N/A FALSE
#> 7 215966560 2018-09-17 N/A FALSE
#> 8 12931053 2014-03-08 within a few hours TRUE
#> 9 116137780 2017-02-12 N/A FALSE
#> 10 113826425 2017-01-29 within a few hours FALSE
#> 11 184607983 2018-04-16 a few days or more FALSE
#> # ... with 17 more variables: host_listings_count <dbl>,
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
#> # price <int>, number_of_reviews <dbl>,
#> # has_availability <lgl>, review_scores_rating <dbl>,
#> # review_scores_accuracy <dbl>, ...
It's pretty clear that there is something odd going on with some of these, but some of them seem
```

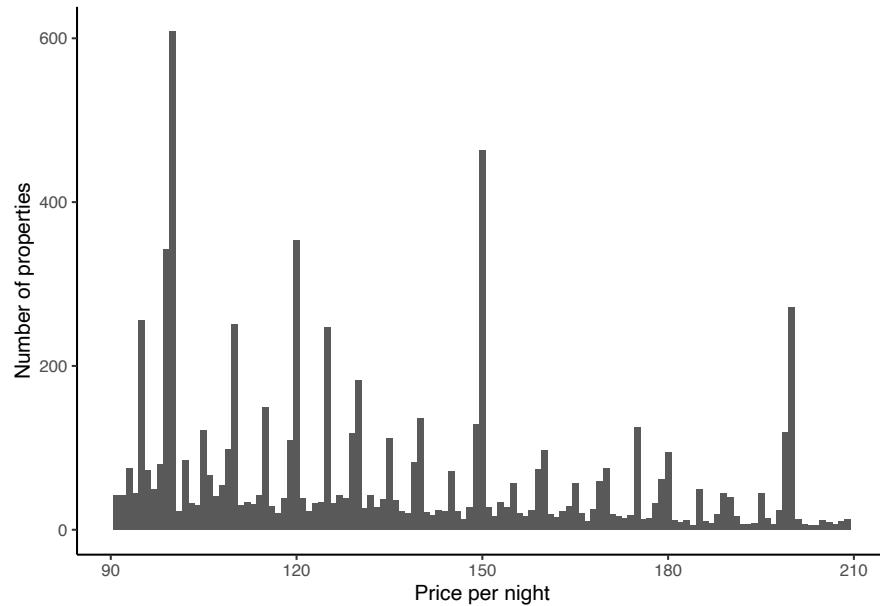
Let's look at the distribution of prices that are in a 'reasonable' range, which until Monica is a full professor, will be defined as a nightly price of less than \$1,000.

```
airbnb_data_selected %>%
 filter(price < 1000) %>%
 ggplot(aes(x = price)) +
 geom_histogram(binwidth = 10) +
 theme_classic() +
 labs(x = "Price per night",
 y = "Number of properties")
```



Interestingly it looks like there is some bunching of prices, possibly around numbers ending in zero or nine? Let's just zoom in on prices between \$90 and \$210, out of interest, but change the bins to be smaller.

```
Look at distribution of price again
airbnb_data_selected %>%
 filter(price > 90) %>%
 filter(price < 210) %>%
 ggplot(aes(x = price)) +
 geom_histogram(binwidth = 1) +
 theme_classic() +
 labs(x = "Price per night",
 y = "Number of properties")
```



#### 14.4.4.2 Superhosts

Airbnb says<sup>6</sup> that:

---

Superhosts are experienced hosts who provide a shining example for other hosts, and extraordinary experiences for their guests.

Once a host reaches Superhost status, a badge superhost badge will automatically appear on their listing and profile to help you identify them.

We check Superhosts' activity four times a year, to ensure that the program highlights the people who are most dedicated to providing outstanding hospitality.

---

First we'll look at the NAs in `host_is_superhost`.

```
airbnb_data_selected %>%
```

---

<sup>6</sup><https://www.airbnb.ca/help/article/828/what-is-a-superhost>

```

 filter(is.na(host_is_superhost))
#> # A tibble: 11 x 21
#> host_id host_since host_response_time host_is_superhost
#> <dbl> <date> <chr> <lgl>
#> 1 23472830 NA <NA> NA
#> 2 31675651 NA <NA> NA
#> 3 75779190 NA <NA> NA
#> 4 47614482 NA <NA> NA
#> 5 201103629 NA <NA> NA
#> 6 111820893 NA <NA> NA
#> 7 23472830 NA <NA> NA
#> 8 196269219 NA <NA> NA
#> 9 23472830 NA <NA> NA
#> 10 266594170 NA <NA> NA
#> 11 118516038 NA <NA> NA
#> # ... with 17 more variables: host_listings_count <dbl>,
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
#> # price <int>, number_of_reviews <dbl>,
#> # has_availability <lgl>, review_scores_rating <dbl>,
#> # review_scores_accuracy <dbl>, ...

```

There are 285 of these and it's clear that there is something odd going on - maybe the host removed the listing or similar?

We'll also want to create a binary variable out of this. It's true/false at the moment, which is fine for the modelling, but there are a handful of situations where it'll be easier if we have a 0/1.

```

airbnb_data_selected <-
 airbnb_data_selected %>%
 mutate(host_is_superhost_binary = case_when(
 host_is_superhost == TRUE ~ 1,
 host_is_superhost == FALSE ~ 0,
 TRUE ~ 999
))
airbnb_data_selected$host_is_superhost_binary[airbnb_data_selected$host_is_superhost_binary == 999]

```

#### 14.4.4.3 Reviews

Airbnb says<sup>7</sup> that:

---

In addition to written reviews, guests can submit an overall star rating and a set of category star ratings for their stay.

Hosts can view their star ratings on their Progress page, under Reviews. Hosts using professional hosting tools can find reviews and quality details on their Performance page, under Quality.

Guests can give ratings on:

- Overall experience. Overall, how was the stay?
- Cleanliness. Did guests feel that the space was clean and tidy?
- Accuracy. How accurately did the listing page represent the space? For example, guests should be able to find up-to-date info and photos in the listing description.
- Value. Did the guest feel that the listing provided good value for the price?
- Communication. How well did you communicate before and during the stay? Guests often care that their host responds quickly, reliably, and frequently to their messages and questions.
- Check-in. How smoothly did check-in go?
- Location. How did guests feel about the neighbourhood? This may mean that there's an accurate description for proximity and access to transportation, shopping centres, city centre, etc., and a description that includes special considerations, like noise, and family safety.
- Amenities. How did guests feel about the amenities that were available during their stay? Guests often care that all the amenities listed are available, working, and in good condition.

In each category, hosts are able to see how often they get 5 stars, how guests rated nearby hosts, and, in some cases, tips to help improve the listing.

The number of stars displayed at the top of a listing page is an aggregate of the primary scores guests have given for that listing. At the bottom of a listing page, there's an aggregate for each category rating. A host needs to receive star ratings from at least 3 guests before their aggregate score appears.

---

<sup>7</sup><https://www.airbnb.ca/help/article/1257/how-do-star-ratings-work-for-stays>

TODO: I don't understand how these review scores are being constructed. Airbnb says it's a star rating, but how are they converting this into the 10 point scale, similarly, how are they constructing the overall one, which seems to be out of 100? There's a lot of clumping around 20, 40, 60, 80, 100 - could they be averaging a five-star scale and then rebasing it to 100?

Now we'll deal with the NAs in `review_scores_rating`. This one is more complicated as there are a lot of them.

```
airbnb_data_selected %>%
 filter(is.na(review_scores_rating))
#> # A tibble: 4,368 x 22
#> host_id host_since host_response_time host_is_superhost
#> <dbl> <date> <chr> <lgl>
#> 1 48239 2009-10-25 N/A FALSE
#> 2 187320 2010-08-01 within a few hours TRUE
#> 3 188183 2010-08-01 a few days or more FALSE
#> 4 187320 2010-08-01 within a few hours TRUE
#> 5 304551 2010-11-29 within an hour TRUE
#> 6 545074 2011-04-29 N/A FALSE
#> 7 1210571 2011-09-26 N/A FALSE
#> 8 1411076 2011-11-15 N/A FALSE
#> 9 1409872 2011-11-15 N/A FALSE
#> 10 1664812 2012-01-28 N/A FALSE
#> # ... with 4,358 more rows, and 18 more variables:
#> # host_listings_count <dbl>,
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
#> # price <int>, number_of_reviews <dbl>,
#> # has_availability <lgl>, review_scores_rating <dbl>, ...
```

Now see if it's just because they don't have any reviews.

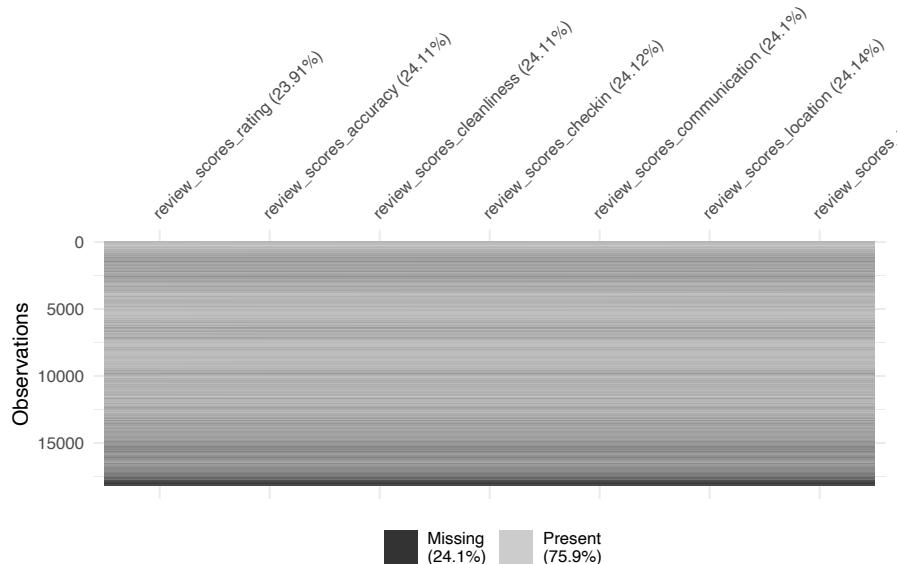
```
airbnb_data_selected %>%
 filter(is.na(review_scores_rating)) %>%
 select(number_of_reviews) %>%
 table()
#> .
#> 0 1 2 3 4
#> 4105 227 23 10 3
```

So it's clear that in almost all these cases they don't have a review yet because they don't have enough reviews. However, it's a large proportion of the

total - almost a fifth of properties don't have any reviews (hence an NA in review\_scores\_rating).

We can use vis\_miss from the visdat package (Tierney, 2017) to make sure that all components of the review are missing. If the NAs are being driven by the Airbnb requirement of at least three reviews then we would expect they would all be missing.

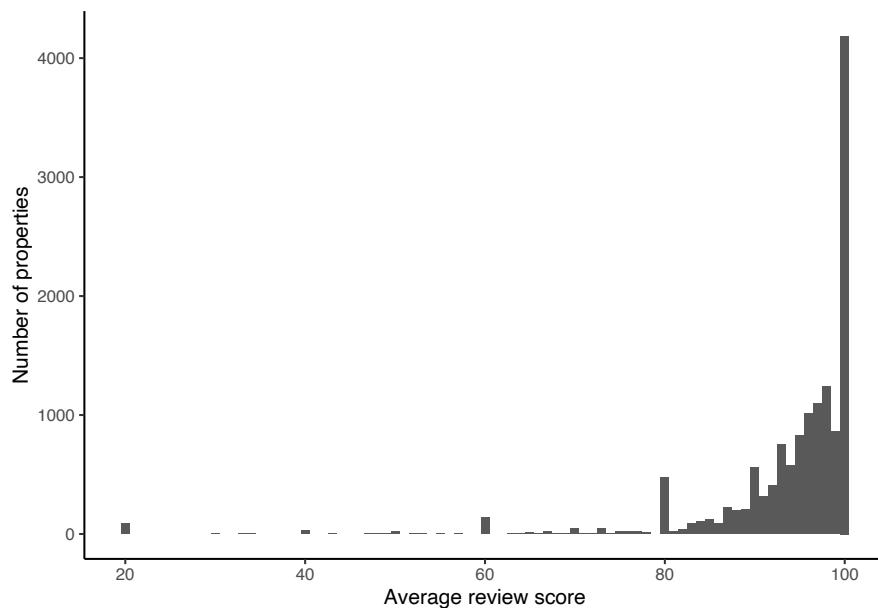
```
We'll just check whether this is the same for all of the different variants of reviews
airbnb_data_selected %>%
 select(review_scores_rating,
 review_scores_accuracy,
 review_scores_cleanliness,
 review_scores_checkin,
 review_scores_communication,
 review_scores_location,
 review_scores_value) %>%
 vis_miss()
```



It looks pretty convincing that in almost all cases, all the different variants of reviews are missing. So let's just focus on the main review.

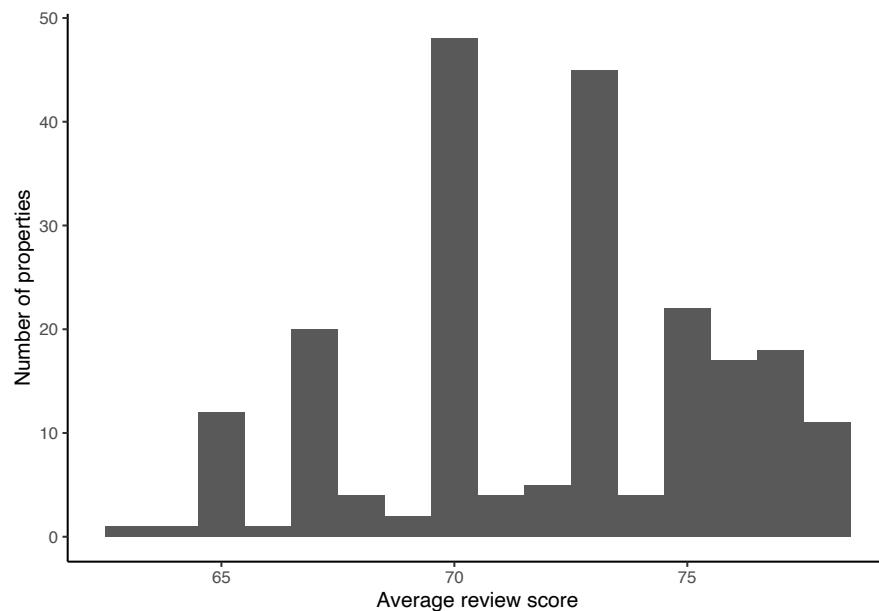
```
airbnb_data_selected %>%
 filter(!is.na(review_scores_rating)) %>%
```

```
ggplot(aes(x = review_scores_rating)) +
 geom_histogram(binwidth = 1) +
 theme_classic() +
 labs(x = "Average review score",
 y = "Number of properties")
```



It's pretty clear that almost all the reviews are more than 80. Let's just zoom in on that 60 to 80 range to check what the distribution looks like in that range.

```
airbnb_data_selected %>%
 filter(!is.na(review_scores_rating)) %>%
 filter(review_scores_rating > 60) %>%
 filter(review_scores_rating < 80) %>%
 ggplot(aes(x = review_scores_rating)) +
 geom_histogram(binwidth = 1) +
 theme_classic() +
 labs(x = "Average review score",
 y = "Number of properties")
```



#### 14.4.4.4 Response time

Airbnb says<sup>8</sup> that:

---

Hosts have 24 hours to officially accept or decline reservation requests. You'll be updated by email about the status of your request.

More than half of all reservation requests are accepted within one hour of being received. The vast majority of hosts reply within 12 hours.

If a host confirms your request, your payment is processed and collected by Airbnb in full. If a host declines your request or the request expires, we don't process your payment.

---

TODO: I don't understand how you can get a response time of NA? It must be related to some other variable.

<sup>8</sup><https://www.airbnb.ca/help/article/75/how-much-time-does-a-host-have-to-respond-to-my-reservation-request>

Looking now at response time:

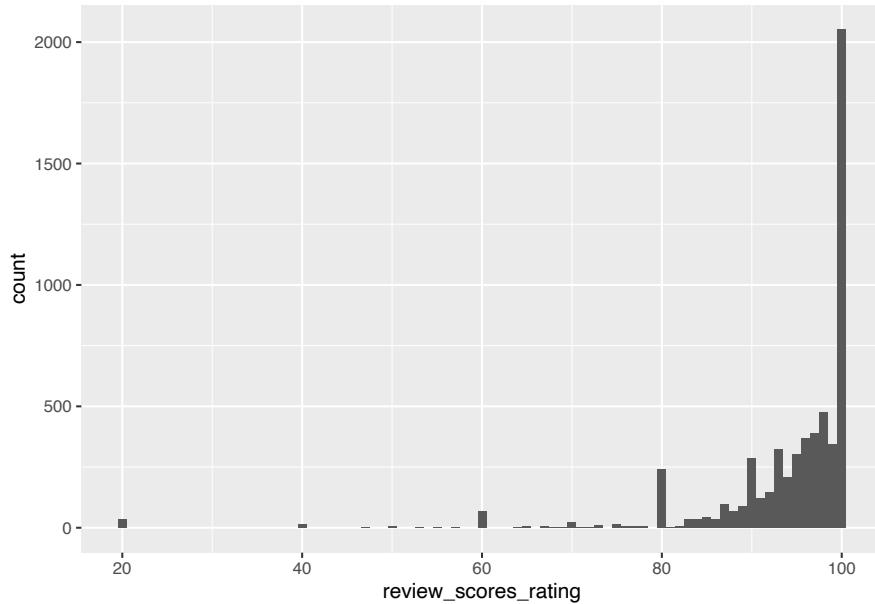
```
table(airbnb_data_selected$host_response_time)
#>
#> a few days or more N/A within a day
#> 816 8469 1235
#> within a few hours within an hour
#> 2062 5672
```

Interestingly it seems like what looks like ‘NAs’ in the `host_response_time` variable are not being coded as proper NAs, but are instead being treated as another category. We’ll recode them to be actual NAs.

```
airbnb_data_selected$host_response_time[airbnb_data_selected$host_response_time == "N/A"] <- NA
```

So here we clearly have issues with NAs. We probably want to filter them away for this example because it’s just a quick example, but there are an awful lot of them (more than 20 per cent) so we’ll have a quick look at them in relation to the review score.

```
airbnb_data_selected %>%
 filter(is.na(host_response_time)) %>%
 ggplot(aes(x = review_scores_rating)) +
 geom_histogram(binwidth = 1)
#> Warning: Removed 2590 rows containing non-finite values
#> (stat_bin).
```



There seem to be an awful lot that have an overall review of 100. There are also an awful lot that have a review score of NA.

```
airbnb_data_selected %>%
 filter(is.na(host_response_time)) %>%
 filter(is.na(review_scores_rating))
#> # A tibble: 2,590 x 22
#> host_id host_since host_response_time host_is_superhost
#> <dbl> <date> <chr> <lgl>
#> 1 48239 2009-10-25 <NA> FALSE
#> 2 545074 2011-04-29 <NA> FALSE
#> 3 1210571 2011-09-26 <NA> FALSE
#> 4 1411076 2011-11-15 <NA> FALSE
#> 5 1409872 2011-11-15 <NA> FALSE
#> 6 1664812 2012-01-28 <NA> FALSE
#> 7 1828773 2012-02-28 <NA> FALSE
#> 8 1923052 2012-03-14 <NA> FALSE
#> 9 2432916 2012-05-22 <NA> FALSE
#> 10 2577688 2012-06-07 <NA> FALSE
#> # ... with 2,580 more rows, and 18 more variables:
#> # host_listings_count <dbl>,
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
```

```
#> # price <int>, number_of_reviews <dbl>,
#> # has_availability <lgl>, review_scores_rating <dbl>, ...
```

#### 14.4.4.5 Host number of listings

There are two versions of a variable telling you how many properties a host has on Airbnb, so to start just check whether there's a difference.

```
airbnb_data_selected %>%
 mutate(listings_count_is_same = if_else(host_listings_count == host_total_listings_count, 1, 0))
 filter(listings_count_is_same == 0)
#> # A tibble: 0 x 23
#> # ... with 23 variables: host_id <dbl>, host_since <date>,
#> # host_response_time <chr>, host_is_superhost <lgl>,
#> # host_listings_count <dbl>,
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
#> # price <int>, number_of_reviews <dbl>, ...
```

There are none in this dataset so we can just remove one column for now and have a quick look at the other one.

```
airbnb_data_selected <-
 airbnb_data_selected %>%
 select(-host_listings_count)

airbnb_data_selected %>%
 count(host_total_listings_count)
#> # A tibble: 49 x 2
#> host_total_listings_count n
#> <dbl> <int>
#> 1 0 2128
#> 2 1 7662
#> 3 2 2476
#> 4 3 1384
#> 5 4 802
#> 6 5 478
#> 7 6 385
#> 8 7 270
#> 9 8 291
#> 10 9 170
#> # ... with 39 more rows
```

So there are a large number who have somewhere in the 2-10 properties range, but the usual long tail. The number with 0 listings is unexpected and worth following up on. And there are a bunch with NA that we'll need to deal with.

```
airbnb_data_selected %>%
 filter(host_total_listings_count == 0) %>%
 head()
#> # A tibble: 6 x 21
#> host_id host_since host_response_time host_is_superhost
#> <dbl> <date> <chr> <lgl>
#> 1 140602 2010-06-08 <NA> FALSE
#> 2 1024550 2011-08-26 <NA> FALSE
#> 3 2647656 2012-06-15 <NA> FALSE
#> 4 3783106 2012-10-06 within an hour FALSE
#> 5 3814089 2012-10-09 within an hour FALSE
#> 6 3827668 2012-10-10 within a day FALSE
#> # ... with 17 more variables:
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
#> # price <int>, number_of_reviews <dbl>,
#> # has_availability <lgl>, review_scores_rating <dbl>,
#> # review_scores_accuracy <dbl>, ...
```

There's nothing that immediately jumps out as odd about the people with zero listings, but there must be something going on.

Based on this dataset, there's a third way of looking at the number of properties someone has and that's to look at the number of times the unique ID occurs.

```
airbnb_data_selected %>%
 count(host_id) %>%
 arrange(-n) %>%
 head()
#> # A tibble: 6 x 2
#> host_id n
#> <dbl> <int>
#> 1 10202618 74
#> 2 1919294 63
#> 3 152088065 59
#> 4 293274089 56
#> 5 785826 54
#> 6 846505 46
```

Again this makes it clear that there are many with multiple properties listed.

#### 14.4.4.6 Decisions

The purpose of this document is to just give a quick introduction to using real-world data, so we'll just remove anything that is annoying, but if you're using this for research then you'd need to justify these decisions and/or possibly make different ones.

Get rid of prices more than \$999.

```
airbnb_data_filtered <-
 airbnb_data_selected %>%
 filter(price < 1000)
dim(airbnb_data_filtered)
#> [1] 18120 21
```

Get rid of anyone with an NA for whether they are a super host.

```
Just remove host_is_superhost NAs for now.
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 filter(!is.na(host_is_superhost))
dim(airbnb_data_filtered)
#> [1] 18109 21
```

Get rid of anyone with an NA in their main review score - this removes roughly 20 per cent of observations.

```
We'll just get rid of them for now, but this is probably something that deserves more attention
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 filter(!is.na(review_scores_rating))
There are still some where the rest of the reviews are missing even though there is a main review
There seem to be an awful lot that have an overall review of 100. Does that make sense?
dim(airbnb_data_filtered)
#> [1] 13801 21
```

Get rid of anyone with a main review score less than 70.

```
We'll just get rid of them for now, but this is probably something that deserves more attention
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 filter(review_scores_rating > 69)
```

```
There are still some where the rest of the reviews are missing even though there is a main review
There seem to be an awful lot that have an overall review of 100. Does that make sense?
dim(airbnb_data_filtered)
#> [1] 13471 21
```

Get rid of anyone with a NA in their response time - this removes roughly another 20 per cent of the observations.

```
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 filter(!is.na(host_response_time))
dim(airbnb_data_filtered)
#> [1] 7763 21
```

TODO: We don't have to do this next step as we've already got rid of them at some other point. So there's something systematic going on and we should come back and look into it.

Get rid of anyone with a NA in their number of properties.

```
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 filter(!is.na(host_total_listings_count))
dim(airbnb_data_filtered)
#> [1] 7763 21
```

Get rid of anyone with a 100 for their review\_scores\_rating.

```
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 filter(review_scores_rating != 100)
dim(airbnb_data_filtered)
#> [1] 5648 21
```

Only keep people with one property:

```
airbnb_data_filtered <-
 airbnb_data_filtered %>%
 add_count(host_id) %>%
 filter(n == 1) %>%
 select(-n)
dim(airbnb_data_filtered)
#> [1] 2304 21
```

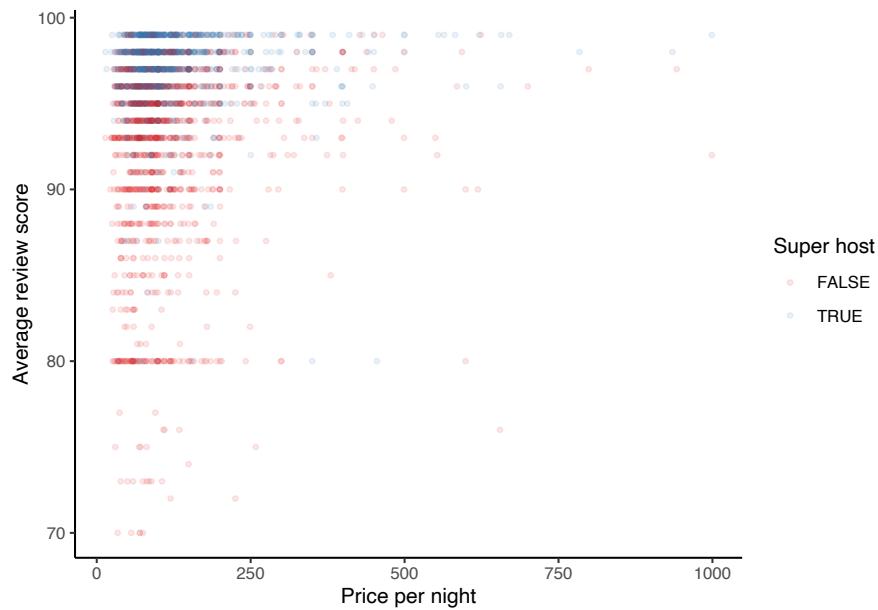
### 14.4.5 Explore data

We might like to make some graphs to see if we can find any relationships jump out. Some aspects that come to mind is looking at prices and reviews and super hosts, and number of properties and neighbourhood.

#### 14.4.5.1 Price and reviews

Look at the relationship between price and reviews, and whether they are a super-host.

```
Look at both price and reviews
airbnb_data_filtered %>%
 ggplot(aes(x = price, y = review_scores_rating, color = host_is_superhost)) +
 geom_point(size = 1, alpha = 0.1) + # Make the points smaller and more transparent as they overlap
 theme_classic() +
 labs(x = "Price per night",
 y = "Average review score",
 color = "Super host") + # Probably should recode this to more meaningful than TRUE/FALSE.
 scale_color_brewer(palette = "Set1")
```



#### 14.4.5.2 Superhost and response-time

One of the aspects that may make someone a super host is how quickly they respond to enquiries. One could imagine that being a superhost involves quickly

saying yes or no to enquiries. Let's look at the data. First, we want to look at the possible values of superhost by their response times.

```
airbnb_data_filtered %>%
 tabyl(host_is_superhost) %>%
 adorn_totals("row") %>%
 adorn_pct_formatting()
#> host_is_superhost n percent
#> FALSE 1224 53.1%
#> TRUE 1080 46.9%
#> Total 2304 100.0%
```

Fortunately, it looks like when we removed the reviews rows we removed any NAs from whether they were a super host, but if we go back and look into that we may need to check again. The `tabyl` function within the `janitor` package (Firke, 2020) would list the NAs if there were any, but in case you don't trust it, another way of check this is to try to filter to just the NAs.

```
airbnb_data_filtered %>%
 filter(is.na(host_is_superhost))
#> # A tibble: 0 x 21
#> # ... with 21 variables: host_id <dbl>, host_since <date>,
#> # host_response_time <chr>, host_is_superhost <lgl>,
#> # host_total_listings_count <dbl>,
#> # host_neighbourhood <chr>, neighbourhood_cleansed <chr>,
#> # room_type <chr>, bathrooms <lgl>, bedrooms <dbl>,
#> # price <int>, number_of_reviews <dbl>,
#> # has_availability <lgl>, review_scores_rating <dbl>, ...
```

Now let's look at the response time.

```
airbnb_data_filtered %>%
 tabyl(host_response_time) %>%
 adorn_totals("row") %>%
 adorn_pct_formatting()
#> host_response_time n percent
#> a few days or more 167 7.2%
#> within a day 378 16.4%
#> within a few hours 519 22.5%
#> within an hour 1240 53.8%
#> Total 2304 100.0%
```

So a vast majority respond within an hour.

Finally, we can look at the cross-tab.

```
airbnb_data_filtered %>%
 tabyl(host_response_time, host_is_superhost) %>%
 adorn_percentages("row") %>%
 adorn_pct_formatting(digits = 0) %>%
 adorn_ns() %>%
 adorn_title()
#> host_is_superhost
#> host_response_time FALSE TRUE
#> a few days or more 88% (147) 12% (20)
#> within a day 67% (254) 33% (124)
#> within a few hours 51% (266) 49% (253)
#> within an hour 45% (557) 55% (683)
```

So if someone doesn't respond within an hour then it's unlikely that they are a super host.

#### 14.4.5.3 Neighbourhood

Finally, let's look at neighbourhood. The data provider has attempted to clean the neighbourhood variable for us, so we'll just use this for now. If we were doing this analysis properly, we'd need to check whether they'd made any mistakes.

```
We expect something in the order of 100 to 150 neighbourhoods, with the top ten accounting for a
airbnb_data_filtered %>%
 tabyl(neighbourhood_cleansed) %>%
 adorn_totals("row") %>%
 adorn_pct_formatting() %>%
 nrow()
#> [1] 140
```

```
airbnb_data_filtered %>%
 tabyl(neighbourhood_cleansed) %>%
 adorn_pct_formatting() %>%
 arrange(-n) %>%
 filter(n > 100) %>%
 adorn_totals("row") %>%
 head()
#> neighbourhood_cleansed n percent
#> Waterfront Communities-The Island 409 17.8%
#> Niagara 101 101 4.4%
#> Total 510 510 -
```

### 14.4.6 Model data

We will now run some models on our dataset. We will first split the data into test/training groups, we do this using functions from the `tidymodels` package ([Kuhn and Wickham, 2020](#)) (which like the `tidyverse` package ([Wickham et al., 2019a](#)) is a package of packages).

```
set.seed(853)

airbnb_data_filtered_split <-
 airbnb_data_filtered %>%
 initial_split(prop = 3/4)

airbnb_train <- training(airbnb_data_filtered_split)
airbnb_test <- testing(airbnb_data_filtered_split)

rm(airbnb_data_filtered_split)
```

#### 14.4.6.1 Logistic regression

We may like to look at whether we can forecast whether someone is a super host, and the factors that go into explaining that. As the dependent variable is binary, this is a good opportunity to look at logistic regression. We expect that better reviews will be associated with faster responses and higher reviews. Specifically, the model that we estimate is:

$$\text{Prob}(\text{Is super host} = 1) = \beta_0 + \beta_1 \text{Response time} + \beta_2 \text{Reviews} + \epsilon$$

We estimate the model using `glm` in the R language ([R Core Team, 2021](#)).

```
logistic_reg_superhost_response_review <- glm(host_is_superhost ~
 host_response_time +
 review_scores_rating,
 data = airbnb_train,
 family = binomial
)
```

We can have a quick look at the results, for instance, the `summary` function. We could also use `tidy` or `glance` from the `broom` package ([Robinson et al., 2020](#)). The details should be the same, but the `broom` functions are tibbles, which means that we can more easily deal with them within a tidy framework.

```
summary(logistic_reg_superhost_response_review)
#>
```

```

#> Call:
#> glm(formula = host_is_superhost ~ host_response_time + review_scores_rating,
#> family = binomial, data = airbnb_train)
#>
#> Deviance Residuals:
#> Min 1Q Median 3Q Max
#> -1.9297 -0.8873 -0.0416 0.8310 4.0657
#>
#> Coefficients:
#> Estimate Std. Error
#> (Intercept) -40.5578 2.4208
#> host_response_timewithin a day 1.5351 0.3466
#> host_response_timewithin a few hours 1.9520 0.3360
#> host_response_timewithin an hour 2.2883 0.3240
#> review_scores_rating 0.4037 0.0248
#> z value Pr(>|z|)
#> (Intercept) -16.754 < 2e-16 ***
#> host_response_timewithin a day 4.429 9.46e-06 ***
#> host_response_timewithin a few hours 5.809 6.28e-09 ***
#> host_response_timewithin an hour 7.062 1.65e-12 ***
#> review_scores_rating 16.276 < 2e-16 ***
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#> Null deviance: 2392.2 on 1727 degrees of freedom
#> Residual deviance: 1750.0 on 1723 degrees of freedom
#> AIC: 1760
#>
#> Number of Fisher Scoring iterations: 6
tidy(logistic_reg_superhost_response_review)
#> # A tibble: 5 x 5
#> term estimate std.error statistic p.value
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 (Intercept) -40.6 2.42 -16.8 5.31e-63
#> 2 host_response_time~ 1.54 0.347 4.43 9.46e- 6
#> 3 host_response_time~ 1.95 0.336 5.81 6.28e- 9
#> 4 host_response_time~ 2.29 0.324 7.06 1.65e-12
#> 5 review_scores_rating 0.404 0.0248 16.3 1.45e-59
glance(logistic_reg_superhost_response_review)
#> # A tibble: 1 x 8
#> null.deviance df.null logLik AIC BIC deviance
#> null.deviance df.null logLik AIC BIC deviance

```

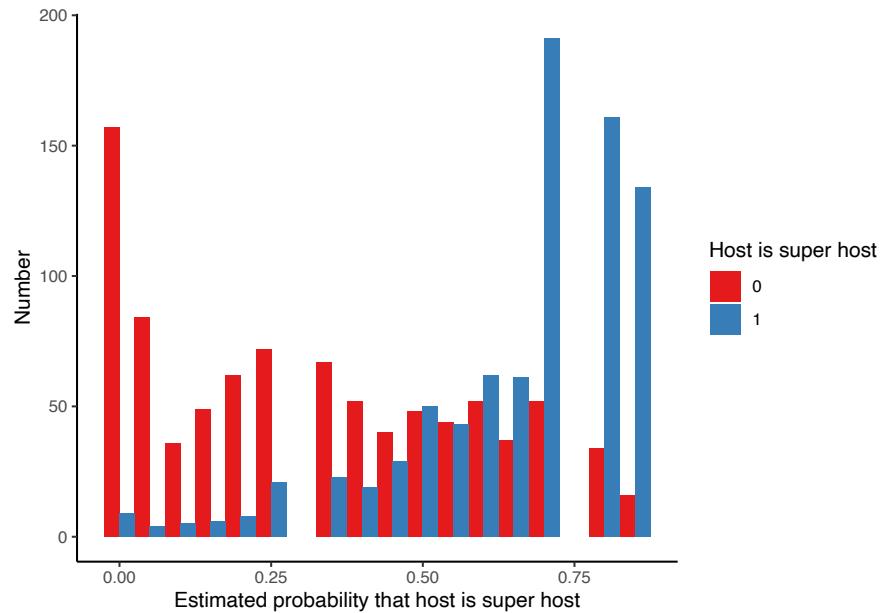
```
#> <dbl> <int> <dbl> <dbl> <dbl> <dbl>
#> 1 2392. 1727 -875. 1760. 1787. 1750.
#> # ... with 2 more variables: df.residual <int>, nobs <int>
```

We might like to look at what our model predicts, compared with whether the person was actually a super host. We can do that in a variety of ways, but one way is to use `augment` from the `broom` package (Robinson et al., 2020). This will add the prediction and associated uncertainty to the data. For every row we will then have the probability that our model is estimating that they are a superhost. But ultimately, we need a binary forecast. There are a bunch of different options, but one is to just say that if the model estimates a probability of more than 0.5 then we bin it into a superhost, and other not.

```
airbnb_data_filtered_logistic_fit_train <-
 augment(logistic_reg_superhost_response_review,
 data = airbnb_train %>% select(host_is_superhost,
 host_is_superhost_binary,
 host_response_time,
 review_scores_rating
),
 type.predict = "response") %>% # We use the "response" option here so that the function
 select(-.hat, -.sigma, -.cooks_d, -.std.resid) %>%
 mutate(predict_host_is_superhost = if_else(.fitted > 0.5, 1, 0), # How do things change if we change
 host_is_superhost_binary = as.factor(host_is_superhost_binary),
 predict_host_is_superhost_binary = as.factor(predict_host_is_superhost)
)
```

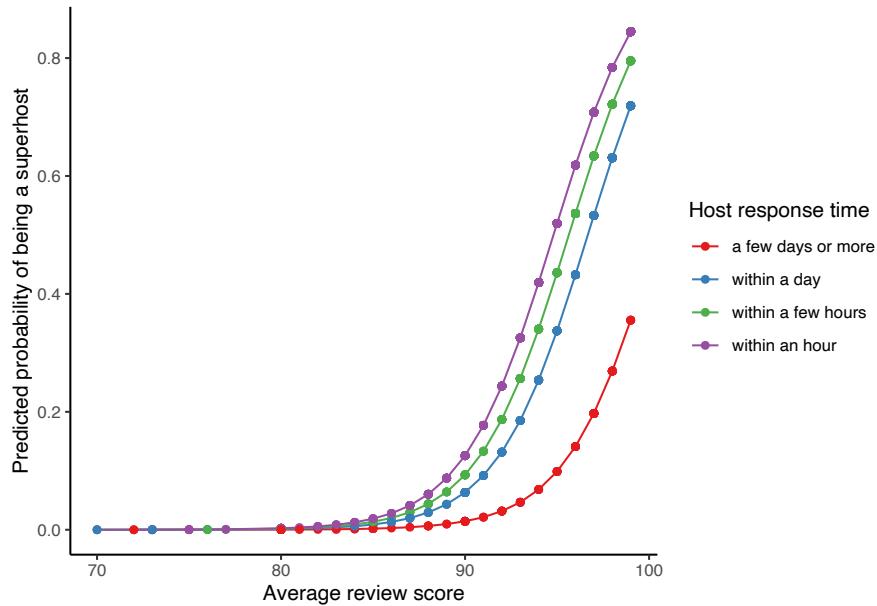
We can look at how far off the model is. There are a bunch of ways of doing this, but one is to look at what probability the model has given each person.

```
airbnb_data_filtered_logistic_fit_train %>%
 ggplot(aes(x = .fitted, fill = host_is_superhost_binary)) +
 geom_histogram(binwidth = 0.05, position = "dodge") +
 theme_classic() +
 labs(x = "Estimated probability that host is super host",
 y = "Number",
 fill = "Host is super host") +
 scale_fill_brewer(palette = "Set1")
```



We can look at how the model probabilities change based on average review score, and their average time to respond.

```
ggplot(airbnb_data_filtered_logistic_fit_train,
 aes(x = review_scores_rating,
 y = .fitted,
 color = host_response_time)) +
 geom_line() +
 geom_point() +
 labs(x = "Average review score",
 y = "Predicted probability of being a superhost",
 color = "Host response time") +
 theme_classic() +
 scale_color_brewer(palette = "Set1")
```



This nice thing about this graph is that it illustrates nicely the effect of a host having an average response time of, say, ‘within an hour’ compared with ‘within a few hours’.

We can focus on how the model does in terms of raw classification using `confusionMatrix` from the `caret` package (Kuhn, 2020). This also gives a bunch of diagnostics (the help file explains what they are). In general, they suggest this isn’t the best model that’s ever existed.

```

caret::confusionMatrix(data = airbnb_data_filtered_logistic_fit_train$predict_host_is_superhost_binary,
 reference = airbnb_data_filtered_logistic_fit_train$host_is_superhost_binary)
#> Confusion Matrix and Statistics
#>
#> Reference
#> Prediction 0 1
#> 0 619 124
#> 1 283 702
#>
#> Accuracy : 0.7645
#> 95% CI : (0.7437, 0.7843)
#> No Information Rate : 0.522
#> P-Value [Acc > NIR] : < 2.2e-16
#>
#> Kappa : 0.5318
#>

```

```
#> Mcnemar's Test P-Value : 4.811e-15
#>
#> Sensitivity : 0.6863
#> Specificity : 0.8499
#> Pos Pred Value : 0.8331
#> Neg Pred Value : 0.7127
#> Prevalence : 0.5220
#> Detection Rate : 0.3582
#> Detection Prevalence : 0.4300
#> Balanced Accuracy : 0.7681
#>
#> 'Positive' Class : 0
#>
```

In any case, to this point we've been looking at how the model has done on the training set. It's also relevant how it does on the test set. Again, there are a bunch of ways to do this, but one is to again use the augment function, but to include a newdata argument.

```
airbnb_data_filtered_logistic_fit_test <-
 augment(logistic_reg_superhost_response_review,
 data = airbnb_train %>% select(host_is_superhost,
 host_is_superhost_binary,
 host_response_time,
 review_scores_rating
),
 newdata = airbnb_test %>% select(host_is_superhost,
 host_is_superhost_binary,
 host_response_time,
 review_scores_rating
), # I'm selecting just because the
 # dataset is quite wide, and so this makes it easier to look at.
 type.predict = "response") %>%
 mutate(predict_host_is_superhost = if_else(.fitted > 0.5, 1, 0),
 host_is_superhost_binary = as.factor(host_is_superhost_binary),
 predict_host_is_superhost_binary = as.factor(predict_host_is_superhost)
)
```

We would expect the performance to be slightly worse on the test set. But it's actually fairly similar.

```
caret::confusionMatrix(data = airbnb_data_filtered_logistic_fit_test$predict_host_is_superhost_bin
 reference = airbnb_data_filtered_logistic_fit_test$host_is_superhost_binary)
```

```
#> Confusion Matrix and Statistics
#>
#> Reference
#> Prediction 0 1
#> 0 197 36
#> 1 125 218
#>
#> Accuracy : 0.7205
#> 95% CI : (0.6819, 0.7568)
#> No Information Rate : 0.559
#> P-Value [Acc > NIR] : 1.018e-15
#>
#> Kappa : 0.4533
#>
#> McNemar's Test P-Value : 4.052e-12
#>
#> Sensitivity : 0.6118
#> Specificity : 0.8583
#> Pos Pred Value : 0.8455
#> Neg Pred Value : 0.6356
#> Prevalence : 0.5590
#> Detection Rate : 0.3420
#> Detection Prevalence : 0.4045
#> Balanced Accuracy : 0.7350
#>
#> 'Positive' Class : 0
#>
```

We could compare the test with the training sets in terms of forecasts.

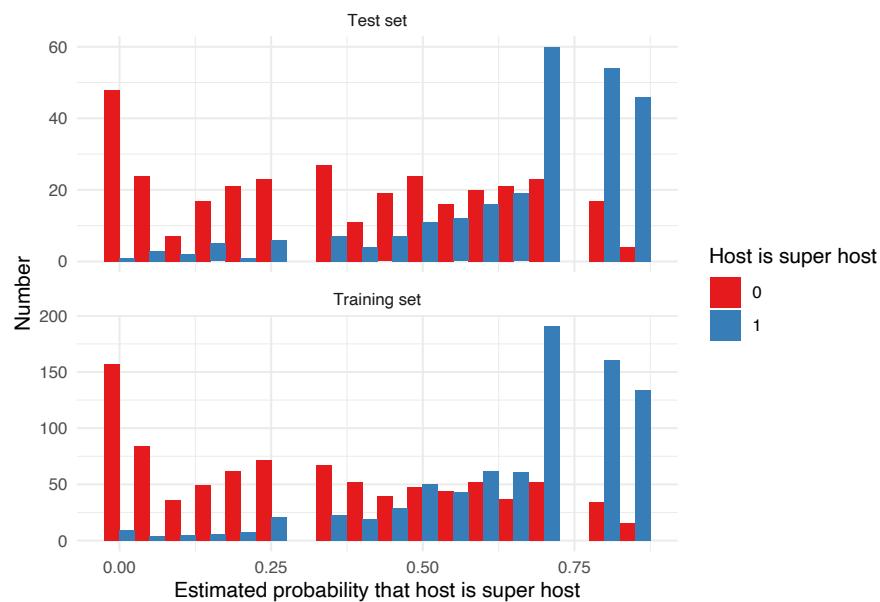
```
training <- airbnb_data_filtered_logistic_fit_train %>%
 select(host_is_superhost_binary, .fitted) %>%
 mutate(type = "Training set")

test <- airbnb_data_filtered_logistic_fit_test %>%
 select(host_is_superhost_binary, .fitted) %>%
 mutate(type = "Test set")

both <- rbind(training, test)
rm(training, test)

both %>%
 ggplot(aes(x = .fitted,
```

```
fill = host_is_superhost_binary)) +
geom_histogram(binwidth = 0.05, position = "dodge") +
theme_minimal() +
labs(x = "Estimated probability that host is super host",
y = "Number",
fill = "Host is super host") +
scale_fill_brewer(palette = "Set1") +
facet_wrap(vars(type),
nrow = 2,
scales = "free_y")
```



---

## 14.5 Exercises and tutorial

### 14.5.1 Exercises

### 14.5.2 Tutorial

# 15

---

## *It's Just A Linear Model*

---

**STATUS:** Under construction.

### Required reading

- Greenland, Sander, Stephen J. Senn, Kenneth J. Rothman, John B. Carlin, Charles Poole, Steven N. Goodman, and Douglas G. Altman, 2016, ‘Statistical tests, P values, confidence intervals, and power: a guide to misinterpretations’, *European journal of epidemiology*, 31, no. 4, pp. 337-350.
- James, Gareth, Daniela Witten, Trevor Hastie and Robert Tibshirani, 2017, *An Introduction to Statistical Learning with Applications in R*, 1st Edition, Chapters 3 and 4.1-4.3., <https://www.statlearning.com>.
- Obermeyer, Z., Powers, B., Vogeli, C., & Sendhill, M., 2019, ‘Dissecting racial bias in an algorithm used to manage the health of populations’, *Science*, (366): 447-453.
- Wickham, Hadley, and Garrett Grolemund, 2017, *R for Data Science*, Chapter 23, <https://r4ds.had.co.nz/>.
- Zook M, Barocas S, boyd d, Crawford K, Keller E, Gangadharan SP, et al. (2017) ‘Ten simple rules for responsible big data research’, *PLoS Comput Biol* 13(3): e1005399. <https://doi.org/10.1371/journal.pcbi.1005399>

### Recommended reading

- Angrist, Joshua D., and Jörn-Steffen Pischke, 2008, *Mostly harmless econometrics: An empiricist's companion*, Princeton University Press, Chapter 3.4.3.
- Cunningham, Scott, *Causal Inference: The Mixtape*, Chapter 2, Yale University Press, <https://mixtape.scunning.com>.
- ElHabr, Tony, 2019, ‘A Bayesian Approach to Ranking English Premier League Teams (using R)’, <https://tonyelhabr.rbind.io/post/bayesian-statistics-english-premier-league/>.
- Ioannidis, John PA, 2005, ‘Why most published research findings are false’, *PLoS medicine*, 2, no. 8, e124.
- Pavlik, Kaylin, 2018, ‘Exploring the Relationship Between Dog Names and Breeds’, <https://www.kaylinpavlik.com/dog-names-tfidf/>.
- Pavlik, Kaylin, 2019, ‘Understanding + classifying genres using Spotify audio features’, <https://www.kaylinpavlik.com/classifying-songs-genres/>.

- Silge, Julia, 2019, ‘Modeling salary and gender in the tech industry’, <https://juliasilge.com/blog/salary-gender/>.
- Silge, Julia, 2019, ‘Opioid prescribing habits in Texas’, <https://juliasilge.com/blog/texas-opioids/>.
- Silge, Julia, 2019, ‘Tidymodels’, <https://juliasilge.com/blog/intro-tidymodels/>.
- Silge, Julia, 2020, ‘#TidyTuesday hotel bookings and recipes’, <https://juliasilge.com/blog/hotels-recipes/>.
- Silge, Julia, 2020, ‘Hyperparameter tuning and #TidyTuesday food consumption’, <https://juliasilge.com/blog/food-hyperparameter-tune/>.
- Taddy, Matt, 2019, *Business Data Science*, Chapters 2 and 4.
- Wasserstein, Ronald L. and Nicole A. Lazar, 2016, ‘The ASA Statement on p-Values: Context, Process, and Purpose’, *The American Statistician*, 70:2, 129-133, DOI: 10.1080/00031305.2016.1154108.

### Fun reading

- Chellel, Kit, 2018, ‘The Gambler Who Cracked the Horse-Racing Code’, *Bloomberg Businessweek*, 3 May, <https://www.bloomberg.com/news/features/2018-05-03/the-gambler-who-cracked-the-horse-racing-code>.

### Key concepts/skills/etc

- Simple and multiple linear regression.
- Logistic and Poisson regression.
- The key role of uncertainty.
- Threats to validity of inferences
- Overfitting.

### Key libraries

- `broom`
- `huxtable`
- `rstanarm`
- `tidymodels`
- `tidyverse`

### Key functions

- `broom::augment()`
- `broom::glance()`
- `broom::tidy()`
- `glm()`
- `huxtable::huxreg()`
- `lm()`
- `parsnip::fit()`
- `parsnip::linear_reg()`
- `parsnip::logistic_reg()`
- `parsnip::set_engine()`

- `poissonreg::poisson_reg()`
- `rnorm()`
- `rpois()`
- `rsample::initial_split()`
- `rsample::testing()`
- `rsample::training()`
- `sample()`
- `set.seed()`
- `summary()`

### Quiz

1. Please write a linear relationship between some response variable, Y, and some predictor, X. What is the intercept term? What is the slope term? What would adding a hat to these indicate?
2. What is the least squares criterion? Similarly, what is RSS and what are we trying to do when we run least squares regression?
3. What is statistical bias?
4. If there were three variables: Snow, Temperature, and Wind, please write R code that would fit a simple linear regression to explain Snow as a function of Temperature and Wind. What do you think about another explanatory variable - daily stock market returns - to your model?
5. According to [Greenland et al. \(2016\)](#), p-values test (pick one)?
  - a. All the assumptions about how the data were generated (the entire model), not just the targeted hypothesis it is supposed to test (such as a null hypothesis).
  - b. Whether the hypothesis targeted for testing is true or not.
  - c. A dichotomy whereby results can be declared ‘statistically significant’.
6. According to [Greenland et al. \(2016\)](#), a p-value may be small because (select all)?
  - a. The targeted hypothesis is false.
  - b. The study protocols were violated.
  - c. It was selected for presentation based on its small size.
7. According to [Obermeyer et al. \(2019\)](#), why does racial bias occur in an algorithm used to guide health decisions in the US (pick one)?
  - a. The algorithm uses health costs as a proxy for health needs.
  - b. The algorithm was trained on Reddit data.
8. When should we use logistic regression (pick one)?
  - a. Continuous dependent variable.
  - b. Binary dependent variable.
  - c. Count dependent variable.
9. I am interested in studying how voting intentions in the recent US presidential election vary by an individual’s income. I set up a lo-

- gistic regression model to study this relationship. In my study, one possible dependent variable would be (pick one)?
- a. Whether the respondent is a US citizen (yes/no)
  - b. The respondent's personal income (high/low)
  - c. Whether the respondent is going to vote for Trump (yes/no)
  - d. Who the respondent voted for in 2016 (Trump/Clinton)
10. I am interested in studying how voting intentions in the recent US presidential election vary by an individual's income. I set up a logistic regression model to study this relationship. In my study, one possible dependent variable would be (pick one)?
    - a. The race of the respondent (white/not white)
    - b. The respondent's marital status (married/not)
    - c. Whether the respondent is registered to vote (yes/no)
    - d. Whether the respondent is going to vote for Biden (yes/no)
  11. Please explain what a p-value is, using only the term itself (i.e. 'p-value') and words that are amongst the 1,000 most common in the English language according to the XKCD Simple Writer - <https://xkcd.com/simplewriter/>. (Please write one or two paragraphs.)
  12. The mean of a Poisson distribution is equal to its?
    - a. Median.
    - b. Standard deviation.
    - c. Variance.

---

## 15.1 Overview

---

Words! Mere words! How terrible they were! How clear, and vivid, and cruel! One could not escape from them. And yet what a subtle magic there was in them! They seemed to be able to give a plastic form to formless things, and to have a music of their own as sweet as that of viol or of lute. Mere words! Was there anything so real as words?

Oscar Wilde, *The Picture of Dorian Gray*.

---

---

Regression will not sort it out. Regression is indeed an oracle,

but a cruel one. It speaks in riddles and delights in punishing us for asking bad questions.

[McElreath \(2020, p. 162\).](#)

---

Linear models have been around for a long time, at least since Galton and many others (some of whom were eugenicists) used linear regression in earnest. The generalized linear model framework came into being, in a formal sense, in the 70s with the seminal folks being Nelder and Wedderburn ([Nelder and Wedderburn, 1972](#)). The idea of generalized linear models is that we broaden the types of outcomes that are allowed. You're still modelling things as a linear function, but you're not constrained to an outcome that is normally distributed. The outcome can be anything in the exponential family. A further, well, generalization of generalized linear models is generalized additive models where you're not generalizing anything to do with the outcome, but instead the structure of the explanatory side, as it were. We're still explaining the dependent variable as an additive function of bits, but those bits can be functions. This framework, in this way, came about in the 90s, with Hastie and Tibshirani ([Hastie and Tibshirani, 1990](#)) (fun fact, Tibshirani did a stats masters at Toronto, and was a professor here from 1985 through to 1998!).

It's important to recognise that when we build models we are not discovering 'the truth'. We are using the model to help us explore and understand the data that we have. There is no one best model, there are just useful models that help us learn something about the data that we have and hence, hopefully, something about the world from which the data were generated. Ben Rhodes, who was an Obama staffer, titled his White House memoirs 'The World as It Is: A Memoir of the Obama White House'. When we use models, we are similarly trying to understand the world, but as the second part of the title makes clear, there are enormous constraints on the perspective. In the same way that we'd not expect Rhodes to advocate an Australian, Canadian, or even US Republican, perspective about the world, it's silly to expect one model to be universal.

We use models to understand the world. We poke, push, and test them. We build them and rejoice in their beauty, and then seek to understand their limits and ultimately destroy them. It is this process that is important, it is this process that allows us to better understand the world. [McElreath \(2020, p. 19\)](#) talks about small and large worlds, saying '(a)ll statistical modeling has these two frames: the small world of the model itself and the large world we hope to deploy the model in'. To what extent does a model trained on the experiences of straight, cis, men, speak to the world as it is? It's not worthless, but it's also not unimpeachable. To what extent does the model teach us about the data that we have? To what extent do the data that we have reflect the

world for which we would like to draw conclusions? Keep these questions front of mind.

Much of statistics was developed in a vacuum. And that's reasonable because it was developed for situations in which X, Y and Z. The original statisticians were literally able to randomise the order of fields and planting because they literally worked in agricultural stations (CITE). However, almost all subsequent applications have not had those properties. We often teach undergraduates that science proceeds (ADD POINTS ABOUT NULL HYPOTHESIS AND POPPER). If you believe that's how it works, then I have a bridge to sell you. Scientists react to incentives. They dabble, guess, and test, and then follow their guesses and backfill. They apply for grant funding for things that they did last time (because they know that'll work) and then spend the money to conduct other things. All of this is fine. But it's not a world in which a traditional null hypothesis holds, which means p-values and power lose their meaning. While you need to understand the 'old world', you also need to be sophisticated enough to understand when you need to move away from it.

In this chapter we... It is called 'It's Just A Linear Model' after a famous quote by Professor Daniela Witten, who identifies how far we can get with linear models and the huge extent to which they underpin statistics.

## 15.2 Simple linear regression

Source: Mijke Rhemtulla<sup>1</sup>, 3 March 2020.

### 15.2.1 Overview

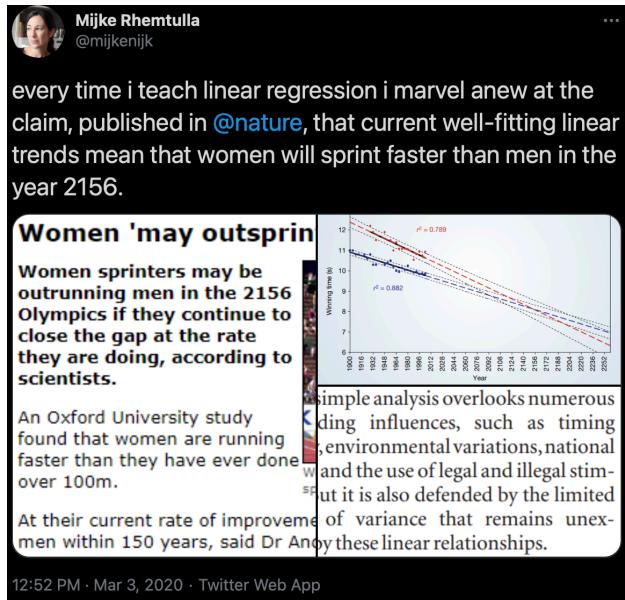
When we have two continuous variables we use simple linear regression. This is based on the Normal (also 'Gaussian') distribution. From Pitman (1993, p. 94) 'The normal distribution with mean  $\mu$  and standard deviation  $\sigma$  is the distribution over the x-axis defined by areas under the normal curve with these parameters. The equation of the normal curve with parameters  $\mu$  and  $\sigma$ , can be written as:

$$y = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}z^2},$$

where  $z = (x - \mu)/\sigma$  measures the number of standard deviations from the mean  $\mu$  to the number  $x$ .

In R we can simulate  $n$  data points from the Normal distribution with `rnorm()`.

<sup>1</sup><https://twitter.com/mijkenijk/status/1234899588311474176>

**FIGURE 15.1:** Oh my.

```
rnorm(n = 20, mean = 0, sd = 1)
#> [1] -1.03830415 0.45007414 -0.92841267 2.07705191
#> [5] 0.01100518 -0.19647894 -0.01035069 0.78473863
#> [9] 0.34374023 0.72619373 -1.25329166 0.05990539
#> [13] 0.03453881 -0.95807034 -0.35676062 -0.08239652
#> [17] 0.28501740 0.09357992 0.20207979 -0.47891842
```

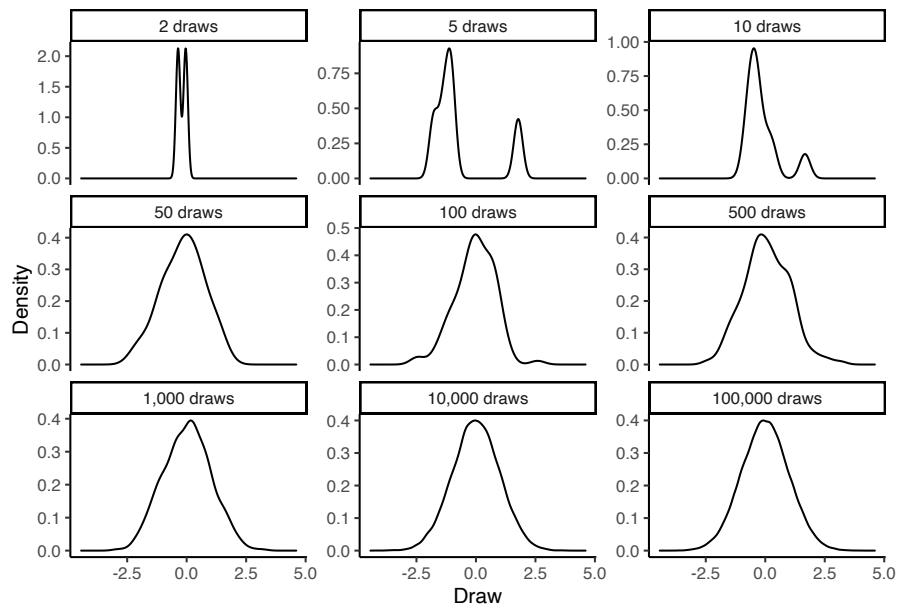
It will take a few draws before we get the expected shape.

```
library(tidyverse)
set.seed(853)
tibble(
 number_of_draws = c(
 rep.int(x = "2 draws", times = 2),
 rep.int(x = "5 draws", times = 5),
 rep.int(x = "10 draws", times = 10),
 rep.int(x = "50 draws", times = 50),
 rep.int(x = "100 draws", times = 100),
 rep.int(x = "500 draws", times = 500),
 rep.int(x = "1,000 draws", times = 1000),
 rep.int(x = "10,000 draws", times = 10000),
```

```

rep.int(x = "100,000 draws", times = 100000),
draws = c(
 rnorm(n = 2, mean = 0, sd = 1),
 rnorm(n = 5, mean = 0, sd = 1),
 rnorm(n = 10, mean = 0, sd = 1),
 rnorm(n = 50, mean = 0, sd = 1),
 rnorm(n = 100, mean = 0, sd = 1),
 rnorm(n = 500, mean = 0, sd = 1),
 rnorm(n = 1000, mean = 0, sd = 1),
 rnorm(n = 10000, mean = 0, sd = 1),
 rnorm(n = 100000, mean = 0, sd = 1)
) %>%
mutate(number_of_draws = as_factor(number_of_draws)) %>%
ggplot(aes(x = draws)) +
geom_density() +
theme_classic() +
facet_wrap(vars(number_of_draws),
scales = "free_y") +
labs(x = 'Draw',
y = 'Density')

```



When we use simple linear regression, we assume that our relationship is characterised by the variables and the parameters, with any difference, of-

ten denoted by  $\epsilon$ , between the expectation and the reality being normally distributed.

If we have two variables,  $Y$  and  $X$ , then we could characterise the relationship between these as:

$$Y \sim \beta_0 + \beta_1 X.$$

There are two coefficients/parameters: the ‘intercept’ is  $\beta_0$ , and the ‘slope’ is  $\beta_1$ . We are saying that  $Y$  will have some value,  $\beta_0$ , even when  $X$  is 0, and that  $Y$  will change by  $\beta_1$  units for every one unit change in  $X$ . The language that we use is that ‘X is being regressed on Y’.

We may then take this relationship to the data that we have about the relationship in order to estimate these coefficients for those particular values that we have:

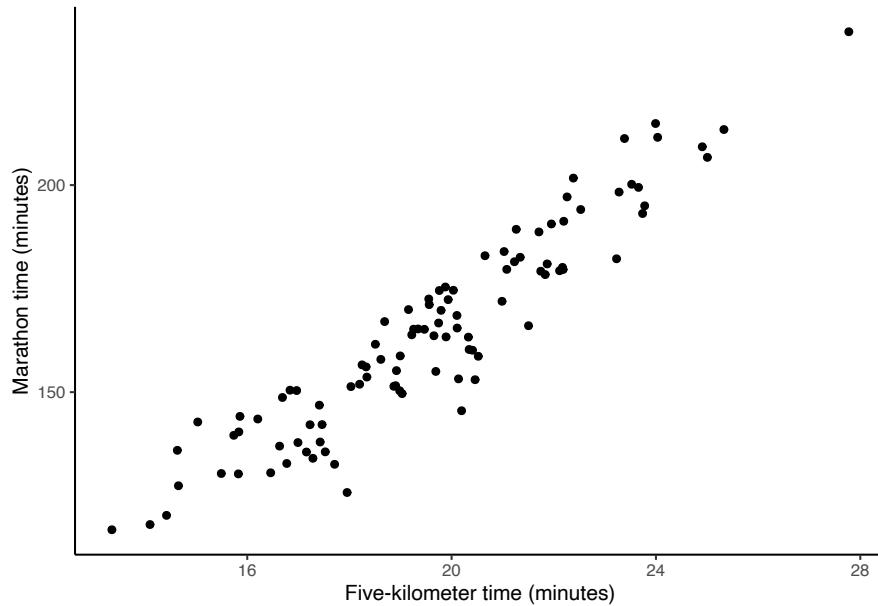
$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x.$$

The hats are used to indicate that these are estimated values. We are saying this is a linear regression because we assume that if  $x$  doubles then  $y$  would also double. Linear regressions considers how the average of a dependent variable changes based on the independent variables.

I want to focus on data, so we’ll make this example concrete, by generating some data and then discussing everything in the context of that. The example will be looking at someone’s time for running five kilometers, compared with their time for running a marathon.

```
set.seed(853)
number_of_observations <- 100
running_data <-
 tibble(five_km_time = rnorm(number_of_observations, 20, 3),
 noise = rnorm(number_of_observations, 0, 10),
 marathon_time = five_km_time * 8.4 + noise,
 was_raining = sample(c("Yes", "No"),
 size = number_of_observations,
 replace = TRUE,
 prob = c(0.2, 0.8)))
)

running_data %>%
 ggplot(aes(x = five_km_time, y = marathon_time)) +
 geom_point() +
 labs(x = "Five-kilometer time (minutes)",
 y = "Marathon time (minutes)") +
 theme_classic()
```



In this set-up we may like to use  $x$ , which is the five-kilometer time, to produce estimates of  $y$ , which is the marathon time. This would involve also estimating values of  $\beta_0$  and  $\beta_1$ , which is why they have a hat on them.

But how should we estimate the coefficients? Even if we impose a linear relationship there are a lot of options (how many straight lines can you fit on a piece of paper?). But clearly some of the fits are not all that great.

One way we may define being great would be to impose that they be as close as possible to each of the  $x$  and  $y$  combinations that we know. There are a lot of candidates for how we define ‘as close as possible’, but one is to minimise the sum of least squares. To do this we produce our estimates of  $\hat{y}$  based on some estimates of  $\hat{\beta}_0$  and  $\hat{\beta}_1$ , given the  $x$ , and then work out how ‘wrong’, for every point  $i$ , we were:

$$e_i = y_i - \hat{y}_i.$$

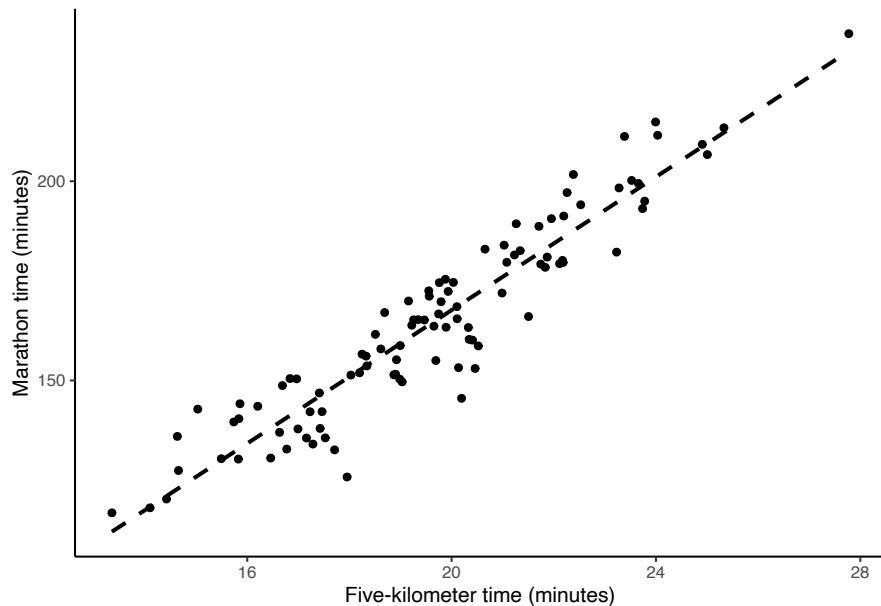
The residual sum of squares (RSS) then requires summing across all the points:  

$$\text{RSS} = e_1^2 + e_2^2 + \dots + e_n^2.$$

This results in one ‘linear best-fit’ line, but it is worth thinking about all of the assumptions and decisions that it took to get us to this point.

```
running_data %>%
 ggplot(aes(x = five_km_time, y = marathon_time)) +
 geom_point() +
 geom_smooth(method = "lm",
```

```
 se = FALSE,
 color = "black",
 linetype = "dashed",
 formula = 'y ~ x') +
 labs(x = "Five-kilometer time (minutes)",
 y = "Marathon time (minutes)") +
 theme_classic()
```



With the least squares criterion we want the values of  $\hat{\beta}_0$  and  $\hat{\beta}_1$  that result in the smallest RSS.

### 15.2.2 Implementation in base R

Within R, the main function for doing linear regression is `lm`. This is included in base R, so you don't need to call any packages, but in a moment, we will call a bunch of packages that will surround `lm` within an environment that we are more familiar with. You specify the relationship with the dependent variable first, then `~`, then the independent variables. Finally, you should specify the dataset (or you could pipe to it as usual).

```
lm(y ~ x, data = dataset)
```

In general, you should assign this to an object:

```
running_data_first_model <-
 lm(marathon_time ~ five_km_time,
 data = running_data)
```

To see the result of your regression you can then call `summary()`.

```
summary(running_data_first_model)
#>
#> Call:
#> lm(formula = marathon_time ~ five_km_time, data = running_data)
#>
#> Residuals:
#> Min 1Q Median 3Q Max
#> -24.763 -5.686 0.722 6.650 16.707
#>
#> Coefficients:
#> Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 0.4114 6.0610 0.068 0.946
#> five_km_time 8.3617 0.3058 27.343 <2e-16 ***
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 8.474 on 98 degrees of freedom
#> Multiple R-squared: 0.8841, Adjusted R-squared: 0.8829
#> F-statistic: 747.6 on 1 and 98 DF, p-value: < 2.2e-16
```

The first part of the result tells us the regression that we called, then information about the residuals, and the estimated coefficients. And then finally some useful diagnostics.

We are considering that there is some relationship between  $X$  and  $Y$ , that is:  $Y = f(X) + \epsilon$ . We are going to say that function,  $f()$ , is linear and so our relationship is:

$$\hat{Y} = \beta_0 + \beta_1 X + \epsilon.$$

There is some ‘true’ relationship between  $X$  and  $Y$ , but we don’t know what it is. All we can do is use our sample of data to try to estimate it. But because our understanding depends on that sample, for every possible sample, we would get a slightly different relationship (as measured by the coefficients).

That  $\epsilon$  is a measure of our error - what does the model not know? There’s going to be plenty that the model doesn’t know, but we hope is that the error does not depend on  $X$ , and that the error is normally distributed.

The intercept is marathon time that we would expect with a five-kilometer

time of 0 minutes. Hopefully this example illustrates the need to carefully interpret the intercept coefficient! The coefficient on five-kilometer run time shows how we expect the marathon time to change if five-kilometer run time changed by one unit. In this case it's about 8.4, which makes sense seeing as a marathon is roughly that many times longer than a five-kilometer run.

### 15.2.3 Tidy up with broom

While there is nothing wrong with the base approach, I want to introduce the `broom` package because that will provide us with outputs in a tidy framework (Robinson et al., 2020). There are three key functions:

- `broom::tidy()`: Gives the coefficient estimates in a tidy output.
- `broom::glance()`: Gives the diagnostics.
- `broom::augment()`: Adds the forecast values, and hence, residuals, to your dataset.

```
library(broom)
tidy(running_data_first_model)
#> # A tibble: 2 x 5
#> term estimate std.error statistic p.value
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 (Intercept) 0.411 6.06 0.0679 9.46e- 1
#> 2 five_km_time 8.36 0.306 27.3 1.17e-47
glance(running_data_first_model)
#> # A tibble: 1 x 12
#> r.squared adj.r.squared sigma statistic p.value df
#> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1 0.884 0.883 8.47 748. 1.17e-47 1
#> # ... with 6 more variables: logLik <dbl>, AIC <dbl>,
#> # BIC <dbl>, deviance <dbl>, df.residual <int>,
#> # nobs <int>
```

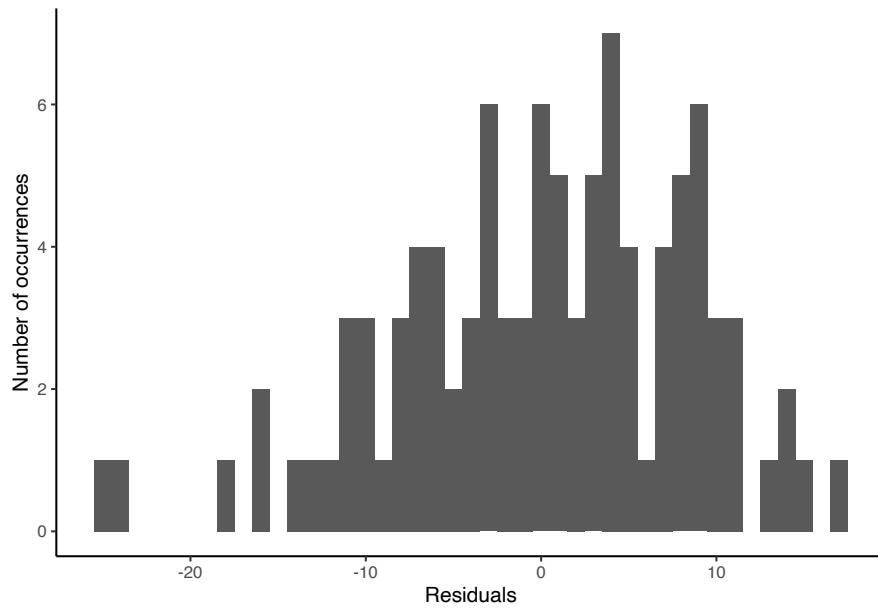
Notice how the results are fairly similar to the base summary function.

```
running_data <-
 augment(running_data_first_model,
 data = running_data)
head(running_data)
#> # A tibble: 6 x 10
#> five_km_time noise marathon_time was_raining .fitted
#> <dbl> <dbl> <dbl> <chr> <dbl>
#> 1 18.9 -3.73 155. No 159.
#> 2 19.9 8.42 175. No 167.
```

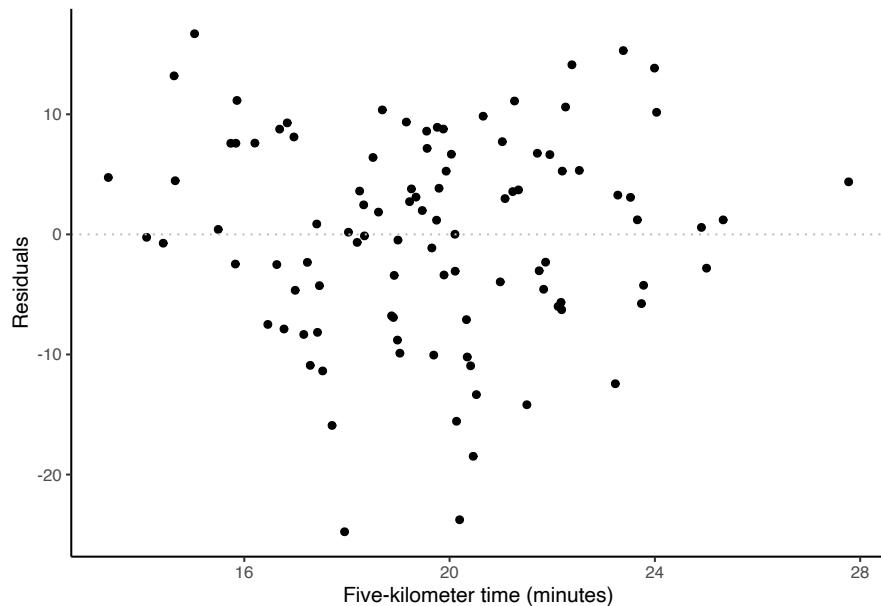
```
#> 3 14.7 4.32 127. No 123.
#> 4 16.6 -2.74 137. No 139.
#> 5 17.0 -4.89 138. No 142.
#> 6 25.3 0.648 213. No 212.
#> # ... with 5 more variables: .resid <dbl>, .hat <dbl>,
#> # .sigma <dbl>, .cooksdi <dbl>, .std.resid <dbl>
```

We could now make plots of the residuals.

```
ggplot(running_data,
 aes(x = .resid)) +
 geom_histogram(binwidth = 1) +
 theme_classic() +
 labs(y = "Number of occurrences",
 x = "Residuals")
```



```
ggplot(running_data, aes(five_km_time, .resid)) +
 geom_point() +
 geom_hline(yintercept = 0, linetype = "dotted", color = "grey") +
 theme_classic() +
 labs(y = "Residuals",
 x = "Five-kilometer time (minutes)")
```



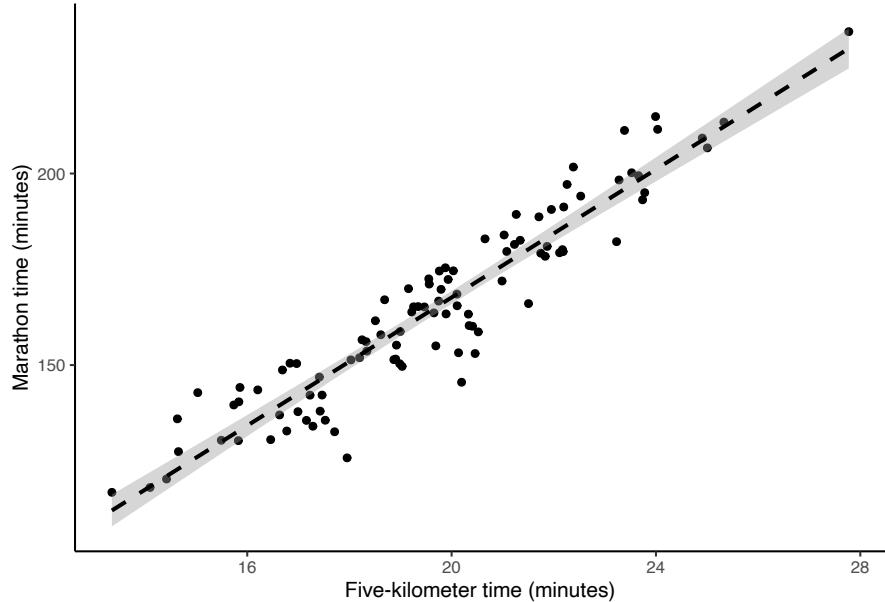
When we say our estimate is unbiased, we are trying to say that even though with some sample our estimate might be too high, and with another sample our estimate might be too low, eventually if we have a lot of data then our estimate would be the same as the population. (A pro hockey player may sometimes shoot right of the net, and sometimes left of the net, but we'd hope that on average they'd be right in the middle of the net). In the words of James et al. (2017), ‘an unbiased estimator does not systematically over- or under-estimate the true parameter’.

But we want to try to speak to the ‘true’ relationship, so we need to try to capture how much we think our understanding depends on the particular sample that we have to analyse. And this is where standard error comes in. It tells us how off our estimate is compared with the actual.

From standard errors, we can compute a confidence interval. A 95 per cent confidence interval means that there is a 0.95 probability that the interval happens to contain the population parameter (which is typically unknown).

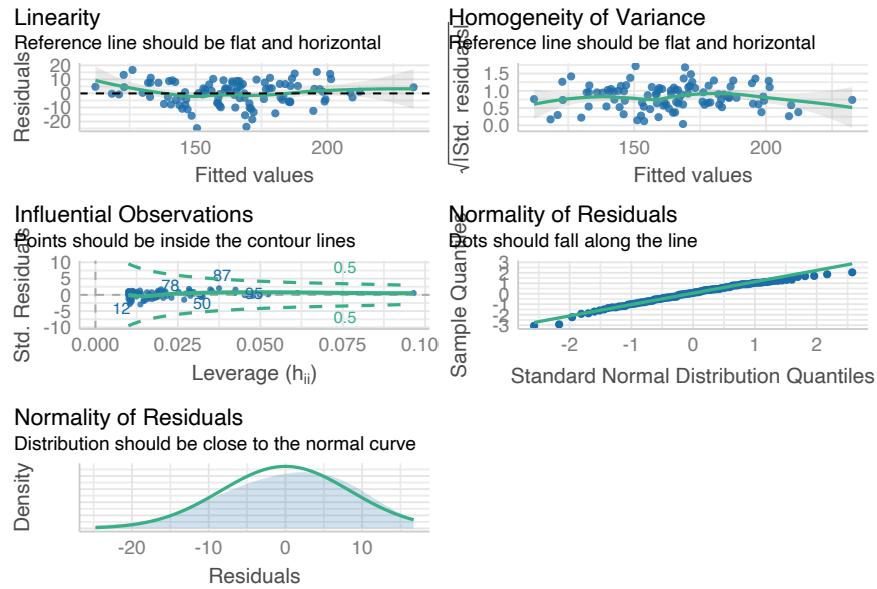
```
running_data %>%
 ggplot(aes(x = five_km_time, y = marathon_time)) +
 geom_point() +
 geom_smooth(method = "lm",
 se = TRUE,
 color = "black",
 linetype = "dashed",
```

```
formula = 'y ~ x') +
 labs(x = "Five-kilometer time (minutes)",
 y = "Marathon time (minutes)") +
 theme_classic()
```



There are a bunch of different tests that you can use to understand how your model is performing given this data. One quick way to look at a whole bunch of different aspects is to use the `performance` package (Lüdecke et al., 2020).

```
library(performance)
performance::check_model(running_data_first_model)
```



#### 15.2.4 Testing hypothesis

Now that we have an interval for which we can say there is a 95 per cent probability it contains the true population parameter we can test claims. For instance, a null hypothesis that there is no relationship between  $X$  and  $Y$  (i.e.  $\beta_1 = 0$ ), compared with an alternative hypothesis that there is some relationship between  $X$  and  $Y$  (i.e.  $\beta_1 \neq 0$ ).

We need to know whether our estimate of  $\beta_1$ , which is  $\hat{\beta}_1$ , is ‘far enough’ away from zero for us to be comfortable claiming that  $\beta_1 \neq 0$ . How far is ‘far enough’? If we were very confident in our estimate of  $\beta_1$  then it wouldn’t have to be far, but if we were not then it would have to be substantial. So it depends on a bunch of things, but essentially the standard error of  $\hat{\beta}_1$ .

We compare this standard error with  $\hat{\beta}_1$  to get the t-statistic:

$$t = \frac{\hat{\beta}_1 - 0}{\text{SE}(\hat{\beta}_1)}.$$

And we then compare our t-statistic to the t-distribution to compute the probability of getting this absolute t-statistic or a larger one, if  $\beta_1 = 0$ . This is the p-value. A small p-value means it is unlikely that we would observe our association due to chance if there wasn’t a relationship.

### 15.2.5 On p-values

The p-value is a specific and subtle concept. It is easy to abuse. The main issue is that it embodies, and assumes correct, every assumption of the model. From Greenland et al. (2016, p. 339): ‘The p-value is then the probability that the chosen test statistic would have been at least as large as its observed value if every model assumption were correct, including the test hypothesis.’ To provide background on the language used here in case you’re unfamiliar, a test hypothesis is typically a ‘null hypothesis’, and a ‘test statistic’ is ‘the distance between the data and the model prediction’ (Greenland et al., 2016).

The following quote (minor edits for consistency with above) summarises the situation:

---

It is true that the smaller the p-value, the more unusual the data would be if every single assumption were correct; but a very small p-value does not tell us which assumption is incorrect. For example, the p-value may be very small because the targeted hypothesis is false; but it may instead (or in addition) be very small because the study protocols were violated, or because it was selected for presentation based on its small size. Conversely, a large p-value indicates only that the data are not unusual under the model, but does not imply that the model or any aspect of it (such as the targeted hypothesis) is correct; it may instead (or in addition) be large because (again) the study protocols were violated, or because it was selected for presentation based on its large size.

The general definition of a p-value may help one to understand why statistical tests tell us much less than what many think they do: Not only does a p-value not tell us whether the hypothesis targeted for testing is true or not; it says nothing specifically related to that hypothesis unless we can be completely assured that every other assumption used for its computation is correct—an assurance that is lacking in far too many studies.

Greenland et al. (2016, p. 339).

---

There is nothing inherently wrong about using p-values, but it is important to use them in sophisticated and thoughtful ways.

Typically one application where it’s easy to see abuse of p-values is in power analysis. As Gelman and Hill (2007, p. 438) say, ‘[s]ample size is never large

enough.... this is not a problem... [w]e are just emphasizing that, just as you never have enough money, because perceived needs increase with resources, your inferential needs with increase with your sample size.' Power refers to the probability of incorrectly failing to reject the null hypothesis. As Imai (2017, p. 303) says:

---

We use power analysis in order to formalize the degree of informativeness of data in hypothesis tests. The power of a statistical hypothesis test is defined as one minus the probability of type II error:

$$\text{power} = 1 - P(\text{type II error})$$

---

In a vacuum, we'd like to have high power and we can achieve that either by having really big effect sizes, or by having a larger number of observations.

---

### 15.3 Multiple linear regression

To this point we've just considered one explanatory variable. But we'll usually have more than one. One approach would be to run separate regressions for each explanatory variable. But compared with separate linear regressions for each, adding more explanatory variables allows us to have a better understanding of the intercept and accounts for interaction. Often the results will be quite different.

---

This slightly counterintuitive result is very common in many real life situations. Consider an absurd example to illustrate the point. Running a regression of shark attacks versus ice cream sales for data collected at a given beach community over a period of time would show a positive relationship, similar to that seen between sales and newspapers. Of course no one (yet) has suggested that ice creams should be banned at beaches to reduce shark attacks. In reality, higher temperatures cause more people to visit the beach, which in turn results in more ice cream sales and more shark attacks. A multiple regression of attacks

versus ice cream sales and temperature reveals that, as intuition implies, the former predictor is no longer significant after adjusting for temperature.

(James et al., 2017, p. 74).

We may also like to consider variables that do not have an inherent ordering. For instance, pregnant or not. When there are only two options then we can use a binary variable which is 0 or 1. If there are more than two levels then use a combination of binary variables, where the ‘missing’ outcome (baseline) gets pushed onto the intercept.

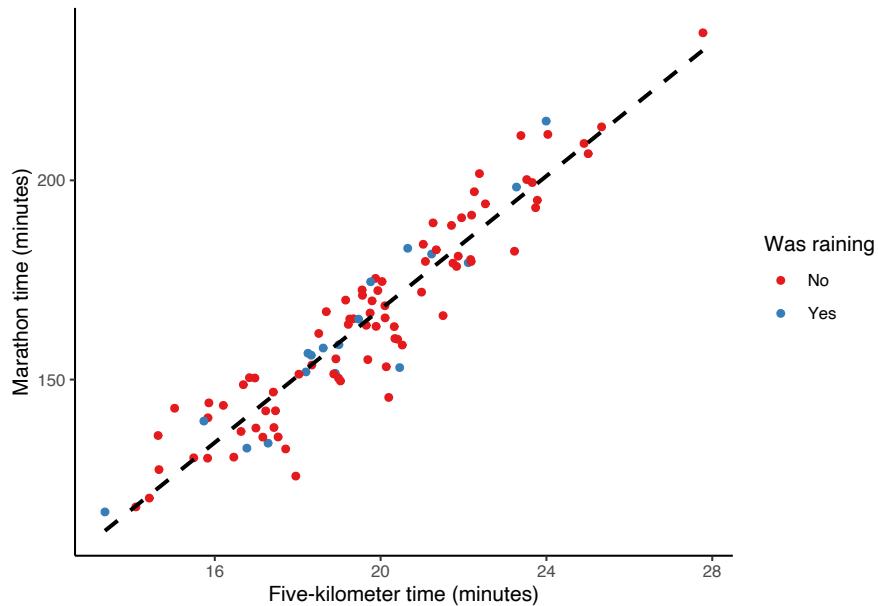
In other languages you may need to explicitly construct dummy variables, but as R was designed as a language to do statistical programming, it does a lot of the work here for you and is fairly forgiving. For instance, if you have a column of character values that only had two values: c("Monica", "Rohan", "Rohan", "Monica", "Monica", "Rohan"), and you used this as a independent variable in your usual regression set up then R would treat it as a dummy variable.

```
running_data_rain_model <-
 lm(marathon_time ~ five_km_time + was_raining,
 data = running_data)
summary(running_data_rain_model)
#>
#> Call:
#> lm(formula = marathon_time ~ five_km_time + was_raining, data = running_data)
#>
#> Residuals:
#> Min 1Q Median 3Q Max
#> -24.6239 -5.5806 0.8377 6.7636 16.8671
#>
#> Coefficients:
#> Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 0.1430 6.1476 0.023 0.981
#> five_km_time 8.3689 0.3081 27.166 <2e-16 ***
#> was_rainingYes 0.7043 2.2220 0.317 0.752
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 8.513 on 97 degrees of freedom
```

```
#> Multiple R-squared: 0.8842, Adjusted R-squared: 0.8818
#> F-statistic: 370.4 on 2 and 97 DF, p-value: < 2.2e-16
```

The result probably isn't too surprising if we look at a plot of the data.

```
running_data %>
 ggplot(aes(x = five_km_time, y = marathon_time, color = was_raining)) +
 geom_point() +
 geom_smooth(method = "lm", se = FALSE, color = "black", linetype = "dashed") +
 labs(x = "Five-kilometer time (minutes)",
 y = "Marathon time (minutes)",
 color = "Was raining") +
 theme_classic() +
 scale_color_brewer(palette = "Set1")
#> `geom_smooth()` using formula 'y ~ x'
```



In addition to wanting to include additional explanatory variables we may think that they are related with one another. For instance, if we were wanting to explain the amount of snowfall in Toronto, then we may be interested in the humidity and the temperature, but those two variables may also interact. We can do this by using `*` instead of `+` when we specify the model in R. If you do interact variables, then you should almost always also include the individual variables as well (Figure 15.2).



**FIGURE 15.2:** Don't leave out the main effects in an interactive model

Source: By Kai Arzheimer<sup>2</sup>, 16 February 2020.

### 15.3.1 Threats to validity and aspects to think about

There are a variety of weaknesses and aspects that you should discuss when you use linear regression. A quick list includes (James et al., 2017, p. 92):

1. Non-linearity of the response-predictor relationships.
2. Correlation of error terms.
3. Non-constant variance of error terms.
4. Outliers.
5. High-leverage points.
6. Collinearity

These are also aspects that you should discuss if you use linear regression. Including plots tends to be handy here to illustrate your points. Other aspects that you may consider discussing include (James et al., 2017, p. 75):

1. Is at least one of the predictors  $X_1, X_2, \dots, X_p$  useful in predicting the response?
2. Do all the predictors help to explain  $Y$ , or is only a subset of the predictors useful?
3. How well does the model fit the data?

<sup>2</sup>[https://twitter.com/kai\\_arzheimer/status/1228998718646607876](https://twitter.com/kai_arzheimer/status/1228998718646607876)

4. Given a set of predictor values, what response value should we predict, and how accurate is our prediction?

### 15.3.2 More credible outputs

Finally, after creating beautiful graphs and tables you may want your regression output to look just as nice. There are a variety of packages in R that will automatically format your regression outputs. One that is particularly nice is `huxtable` (Hugh-Jones, 2020).

```
library(huxtable)
huxreg(running_data_first_model, running_data_rain_model)
```

	(1)	(2)
(Intercept)	0.411	0.143
	(6.061)	(6.148)
five_km_time	8.362 ***	8.369 ***
	(0.306)	(0.308)
was_rainingYes		0.704
		(2.222)
N	100	100
R2	0.884	0.884
logLik	-354.584	-354.532
AIC	715.168	717.064

\*\*\* p < 0.001; \*\* p < 0.01; \* p < 0.05.

### 15.3.3 Implementation in `tidymodels`

The reason that we went to all that trouble to do simple regression is that we often want to fit a bunch of models. One way is to copy/paste code a bunch of times. There's nothing wrong with that. And that's the way that most people get started, but you may want to take an approach that scales more easily.

We also need to think more carefully about over-fitting, and being able to evaluate our models.

The `tidymodels` package (Kuhn and Wickham, 2020) is what all the cool kids are using these days. It's an attempt to bring some order to the chaos that has been different modelling packages in R. (There have been other attempts in the past and they've crashed and burned, but hopefully this time is different.) The issue is that let's say you want to run a simple linear regression and then run a random forest. The language that you'd use to code these models is fairly different. The `tidymodels` package is the latest attempt to bring a coherent grammar to this. It's also a package of packages.

We'll create test and training datasets.

```
set.seed(853)
library(tidymodels)

running_data_split <- rsample::initial_split(running_data, prop = 0.80)
running_data_split
#> #<Analysis/Assess/Total>
#> #<80/20/100>
```

So we have 81 points in our training set, 19 in our test set and 100 in total.

We can then make datasets for the test and training samples.

```
running_data_train <- rsample::training(running_data_split)
running_data_test <- rsample::testing(running_data_split)
```

If we have a look at the dataset that we made we can see that it's got fewer rows. We could have reached the same outcome with something like:

```
running_data <-
 running_data %>%
 mutate(magic_number = sample(x = c(1:nrow(running_data)), size = nrow(running_data), replace = F))

running_data_test <-
 running_data %>%
 filter(magic_number <= 20)

running_data_train <-
 running_data %>%
 filter(magic_number > 20)
```

```
first_go <-
 parsnip::linear_reg() %>%
 parsnip::set_engine(engine = "lm") %>%
 parsnip::fit(marathon_time ~ five_km_time + was_raining,
 data = running_data_train
)
```

### 15.3.4 Implementation in `rstanarm`

The `tidymodels` package will be fine for specific types of tasks. For instance if you are doing machine learning then chances are you are interested in forecasting. That's the kind of thing that `tidymodels` is really built for. If you want equivalent firepower for explanatory modelling then one option is to use Bayesian approaches more directly. Yes, you can use Bayesian models within the `tidymodels` ecosystem, but as you start to move away from out-of-the-box solutions, it becomes important to start to understand what is going on under the hood.

There are a variety of ways of getting started, but essentially what you need is a probabilistic programming language. That is one that is specifically designed for this sort of thing, in comparison to R, which is designed for more general statistical computing. We will use Stan in these notes within the context of our familiar R environment. We will interface with Stan using the `rstanarm` package (Goodrich et al., 2020).

```
library(rstanarm)

first_go_in_rstanarm <-
 stan_lm(
 marathon_time ~ five_km_time + was_raining,
 data = running_data,
 prior = NULL,
 seed = 853
)
```

```
first_go_in_rstanarm
#> stan_lm
#> family: gaussian [identity]
#> formula: marathon_time ~ five_km_time + was_raining
#> observations: 100
#> predictors: 3
#> -----
#>
```

```
#> Median MAD_SD
#> (Intercept) 0.4 6.0
#> five_km_time 8.4 0.3
#> was_rainingYes 0.7 2.2
#>
#> Auxiliary parameter(s):
#> Median MAD_SD
#> R2 0.9 0.0
#> log-fit_ratio 0.0 0.0
#> sigma 8.6 0.6
#>
#> -----
#> * For help interpreting the printed output see ?print.stanreg
#> * For info on the priors used see ?prior_summary.stanreg
```

## 15.4 Logistic regression

### 15.4.1 Overview

To steal a joke from someone, ‘it’s AI when you’re fundraising, machine learning when you’re hiring, and logistic regression when you’re implementing.’

When the dependent variable is a binary outcome, that is 0 or 1, then instead of linear regression we may like to use logistic regression. Although a binary outcome may sound limiting, there are a lot of circumstances in which your outcome either naturally falls into this situation, or can be adjusted into it (e.g. a voter supports the liberals or not the liberals).

The reason that we use logistic regression is that we’ll be modelling a probability and so it will be bounded between 0 and 1. Whereas with linear regression we may end up with values outside this. In practice it is usually fine to start with linear regression and then move to logistic regression as you build confidence.

This all said, logistic regression, as Daniella Witten teaches us, is just a linear model!

### 15.4.2 Implementation in base

I’d like to consider a slightly more interesting example, which is a dataset of pearl jewellery, from the Australian retailer Paspaley.

```

paspaley_dataset <- read_csv("https://raw.githubusercontent.com/RohanAlexander/paspaley/master/out/
#> Rows: 1289 Columns: 13
#> -- Column specification -----
#> Delimiter: ","
#> chr (10): product, name, description, availability, sku, ...
#> dbl (2): price, year
#> lgl (1): keshi
#>
#> i Use `spec()` to retrieve the full column specification for this data.
#> i Specify the column types or set `show_col_types = FALSE` to quiet this message.

paspaley_dataset$metal %>% table()
#> .
#> Other Platinum Rose gold White gold Yellow gold
#> 134 23 89 475 568

```

In this case we'll model whether some jewellery is made of white or yellow gold, based on their price and the year (Figure 15.3).

```

paspaley_logistic_dataset <-
 paspaley_dataset %>%
 filter(metal %in% c('White gold', 'Yellow gold')) %>%
 select(metal, price, year)

```

The graph suggests that we should filter any price higher than \$100,000.

```

paspaley_logistic_dataset <-
 paspaley_logistic_dataset %>%
 filter(price < 100000)

```

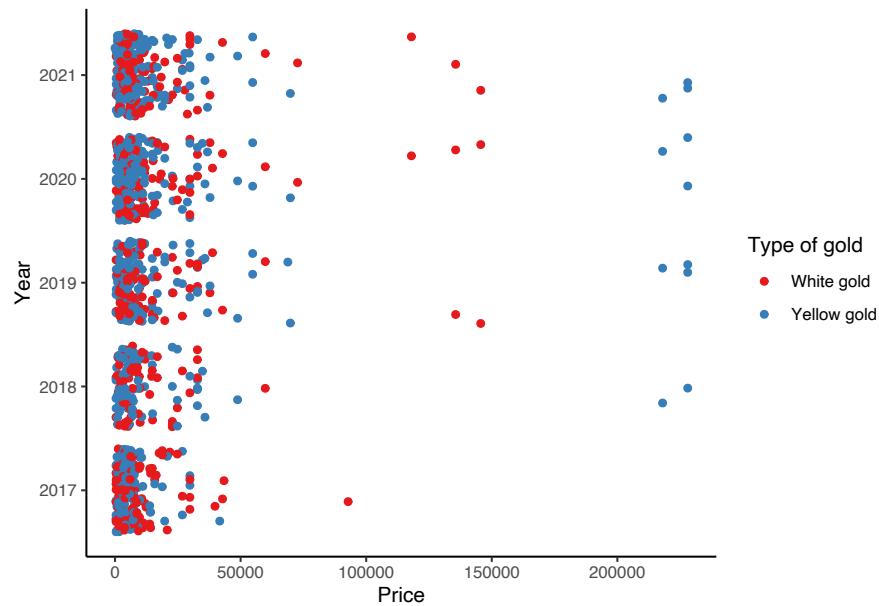
As with linear regression, logistic regression is built into R, with the `glm` function. In this case, we'll try to work out if the jewellery was white gold. Although not strictly necessary for this particular function, we'll change it to a binary, that will be 1 if white gold and 0 if not.

```

paspaley_logistic_dataset <-
 paspaley_logistic_dataset %>%
 mutate(is_white_gold = if_else(metal == "White gold", 1, 0))

white_gold_model <-
 glm(is_white_gold ~ price + year,
 data = paspaley_logistic_dataset,
 family = 'binomial')

```



**FIGURE 15.3:** Examining the type of gold some jewellery is made from.

```
summary(white_gold_model)
#>
#> Call:
#> glm(formula = is_white_gold ~ price + year, family = "binomial",
#> data = paspaley_logistic_dataset)
#>
#> Deviance Residuals:
#> Min 1Q Median 3Q Max
#> -1.250 -1.103 -1.015 1.247 1.353
#>
#> Coefficients:
#> Estimate Std. Error z value Pr(>|z|)
#> (Intercept) 2.087e+02 8.674e+01 2.406 0.0161 *
#> price 3.832e-06 5.405e-06 0.709 0.4783
#> year -1.035e-01 4.296e-02 -2.408 0.0160 *
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
```

```
#>
#> Null deviance: 1411.6 on 1023 degrees of freedom
#> Residual deviance: 1405.5 on 1021 degrees of freedom
#> AIC: 1411.5
#>
#> Number of Fisher Scoring iterations: 4
```

One reason that logistic regression can be a bit of a pain initially is because the coefficients take a bit of work to interpret. In particular, our estimate on price is -3.170e-06. This is the odds. So the odds that it was white gold decrease by -3.170e-06 as the price increases. We can have our model make forecasts in terms of a probability, by asking for that.

```
paspaley_logistic_dataset <-
 broom::augment(white_gold_model,
 data = paspaley_logistic_dataset,
 type.predict = "response")
head(paspaley_logistic_dataset)
```

metal	price	year	is_white_gold	.fitted	.resid	.std.resid	.hat	.sigma	.cooks
Yellow gold	2.58e+03	2.02e+03	0	0.505	-1.19	-1.19	0.0034	1.17	0.00116
Yellow gold	2.08e+03	2.02e+03	0	0.504	-1.18	-1.19	0.00344	1.17	0.00118
Yellow gold	3.08e+03	2.02e+03	0	0.505	-1.19	-1.19	0.00335	1.17	0.00115
White gold	7.38e+03	2.02e+03	1	0.509	1.16	1.16	0.00313	1.17	0.00101
White gold	3.08e+03	2.02e+03	1	0.505	1.17	1.17	0.00335	1.17	0.0011
White gold	3.95e+03	2.02e+03	1	0.506	1.17	1.17	0.00329	1.17	0.00108

### 15.4.3 Implementation in `tidymodels`

We can use `tidymodels` to run this if we wanted. In this case, we need it as a factor.

```
set.seed(853)

paspaley_logistic_dataset <-
```

```

paspaley_logistic_dataset %>%
 mutate(is_white_gold = as_factor(is_white_gold))

paspaley_logistic_dataset_split <- rsample::initial_split(paspaley_logistic_dataset, prop = 0.80)
paspaley_logistic_dataset_train <- rsample::training(paspaley_logistic_dataset_split)
paspaley_logistic_dataset_test <- rsample::testing(paspaley_logistic_dataset_split)

white_gold_model_tidymodels <-
 parsnip::logistic_reg(mode = "classification") %>%
 parsnip::set_engine("glm") %>%
 fit(is_white_gold ~ price + year,
 data = paspaley_logistic_dataset_train)

white_gold_model_tidymodels
#> parsnip model object
#>
#> Fit time: 3ms
#>
#> Call: stats::glm(formula = is_white_gold ~ price + year, family = stats::binomial,
#> data = data)
#>
#> Coefficients:
#> (Intercept) price year
#> 1.832e+02 5.245e-06 -9.082e-02
#>
#> Degrees of Freedom: 818 Total (i.e. Null); 816 Residual
#> Null Deviance: 1130
#> Residual Deviance: 1125 AIC: 1131

```

#### 15.4.4 Implementation in `rstanarm`

```

paspaley_in_rstanarm <-
 rstanarm::stan_glm(
 is_white_gold ~ price + year,
 data = paspaley_logistic_dataset,
 family = binomial(link = "logit"),
 prior = NULL,
 seed = 853
)

```

## 15.5 Poisson regression

### 15.5.1 Overview

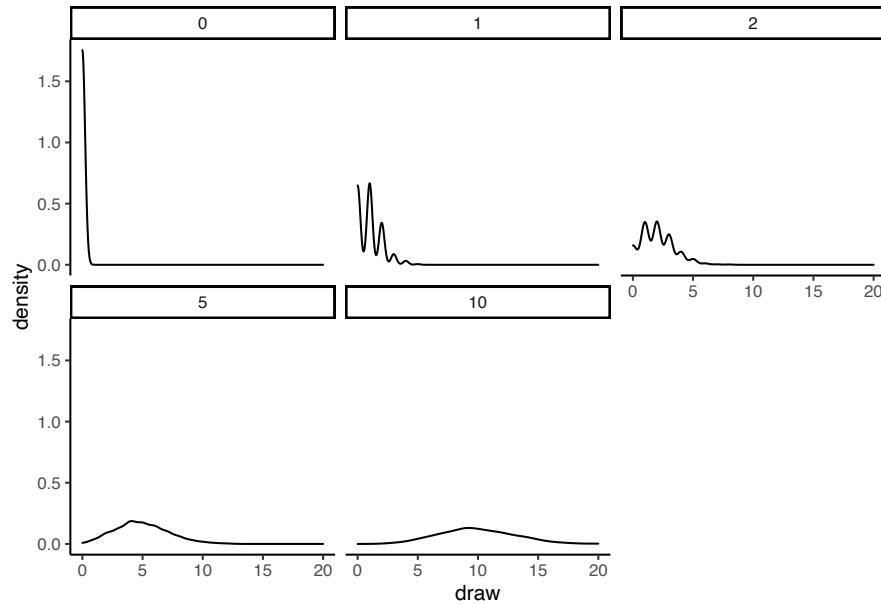
When we have count data, we use Poisson distribution. From Pitman (1993, p. 121) 'The Poisson distribution with parameter  $\mu$  or Poisson ( $\mu$ ) distribution is the distribution of probabilities  $P_\mu(k)$  over  $0, 1, 2, \dots$  defined by:  
$$P_\mu(k) = e^{-\mu} \mu^k / k!, \text{ for } k = 0, 1, 2, \dots$$

We can simulate  $n$  data points from the Poisson distribution with `rpois()` where  $\lambda$  is the mean and the variance.

```
rpois(n = 20, lambda = 3)
#> [1] 2 2 3 4 2 2 4 2 4 1 3 2 3 3 2 2 0 1 2 1
```

That  $\lambda$  parameter governs the shape of the distribution.

```
set.seed(853)
number_of_each <- 1000
tibble(lambda = c(rep(0, number_of_each), rep(1, number_of_each), rep(2, number_of_each), rep(5, n
 draw = c(rpois(n = number_of_each, lambda = 0), rpois(n = number_of_each, lambda = 1), rpoi
 ggplot(aes(x = draw)) +
 geom_density() +
 facet_wrap(vars(lambda)) +
 theme_classic()
```



For instance, if we look at the number of A+ grades that are awarded in each university course in a given term then for each course we would have a count.

```
set.seed(853)
count_of_A_plus <-
 tibble(
 # https://stackoverflow.com/questions/1439513/creating-a-sequential-list-of-letters-with-r
 department = c(rep.int("1", 26), rep.int("2", 26)),
 course = c(paste0("DEP_1_", letters), paste0("DEP_2_", letters)),
 number_of_A_plus = c(sample(c(1:10),
 size = 26,
 replace = TRUE),
 sample(c(1:50),
 size = 26,
 replace = TRUE))
)
)
```

### 15.5.2 Implementation in base

```
grades_model <-
 glm(number_of_A_plus ~ department,
```

```

data = count_of_A_plus,
family = 'poisson')

summary(grades_model)
#>
#> Call:
#> glm(formula = number_of_A_plus ~ department, family = "poisson",
#> data = count_of_A_plus)
#>
#> Deviance Residuals:
#> Min 1Q Median 3Q Max
#> -6.7386 -1.2102 -0.2515 1.3292 3.9520
#>
#> Coefficients:
#> Estimate Std. Error z value Pr(>|z|)
#> (Intercept) 1.44238 0.09535 15.13 <2e-16 ***
#> department2 1.85345 0.10254 18.07 <2e-16 ***
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for poisson family taken to be 1)
#>
#> Null deviance: 816.08 on 51 degrees of freedom
#> Residual deviance: 334.57 on 50 degrees of freedom
#> AIC: 545.38
#>
#> Number of Fisher Scoring iterations: 5

```

### 15.5.3 Implementation in `tidymodels`

We can use `tidymodels` to run this if we wanted although we first need to install a helper package `poissonreg`.

```

install.packages("poissonreg")

set.seed(853)

count_of_A_plus_split <- rsample::initial_split(count_of_A_plus, prop = 0.80)
count_of_A_plus_train <- rsample::training(count_of_A_plus_split)
count_of_A_plus_test <- rsample::testing(count_of_A_plus_split)

a_plus_model_tidymodels <-

```

```
poissonreg::poisson_reg(mode = "regression") %>%
 parsnip::set_engine("glm") %>%
 parsnip::fit(number_of_A_plus ~ department,
 data = count_of_A_plus_train)

a_plus_model_tidymodels
#> parsnip model object
#>
#> Fit time: 3ms
#>
#> Call: stats::glm(formula = number_of_A_plus ~ department, family = stats::poisson,
#> data = data)
#>
#> Coefficients:
#> (Intercept) department2
#> 1.488 1.867
#>
#> Degrees of Freedom: 40 Total (i.e. Null); 39 Residual
#> Null Deviance: 618.6
#> Residual Deviance: 210.1 AIC: 380.4
```

---

## 15.6 Exercises and tutorial

### 15.6.1 Exercises

### 15.6.2 Tutorial

# 16

---

## *Causality from observational data*

---

**STATUS:** Under construction.

**TODO:** Replace the arm matching with <https://kosukeimai.github.io/MatchIt/index.html>

### Required reading

- Angelucci, Charles, and Julia Cagé, 2019, ‘Newspapers in times of low advertising revenues’, *American Economic Journal: Microeconomics*, vol. 11, no. 3, pp. 319-364, DOI: 10.1257/mic.20170306, available at: <https://www.aeaweb.org/articles?id=10.1257/mic.20170306>.
- Better Evaluation, ‘Regression Discontinuity’, <https://www.betterevaluation.org/en/evaluation-options/regressiondiscontinuity>
- Dagan, Noa, Noam Barda, Eldad Kepten, Oren Miron, Shay Perchik, Mark A. Katz, Miguel A. Hernán, Marc Lipsitch, Ben Reis, and Ran D. Balicer, 2021, ‘BNT162b2 mRNA Covid-19 vaccine in a nationwide mass vaccination setting’, *New England Journal of Medicine*, 24 February, <https://www.nejm.org/doi/full/10.1056/NEJMoa2101765>.
- Eggers, Andrew C., Anthony Fowler, Jens Hainmueller, Andrew B. Hall, and James M. Snyder Jr, 2015, ‘On the validity of the regression discontinuity design for estimating electoral effects: New evidence from over 40,000 close races’, *American Journal of Political Science*, 59 (1), pp. 259-274
- Gelman, Andrew, 2019, ‘Another Regression Discontinuity Disaster and what can we learn from it’, 25 June, <https://statmodeling.stat.columbia.edu/2019/06/25/another-regression-discontinuity-disaster-and-what-can-we-learn-from-it/>.
- Gelman, Andrew, Jennifer Hill and Aki Vehtari, 2020, *Regression and Other Stories*, Cambridge University Press, Chs 18 - 21.
- Gertler, Paul, Sebastian Martinez, Patrick Premand, Laura Rawlings, and Christel Vermeersch, ‘Impact Evaluation in Practice’, Chapter 5 - 8.
- McElreath, Richard, 2020, *Statistical Rethinking*, 2nd Edition, CRC Press, Ch 14.
- Meng, Xiao-Li, 2021, ‘What Are the Values of Data, Data Science, or Data Scientists?’, *Harvard Data Science Review*, <https://doi.org/10.1162/99608f92.ee717cf7>, <https://hdsr.mitpress.mit.edu/pub/bj2dfcwg/release/2>.

- Riederer, Emily, 2021, ‘Causal design patterns for data analysts’, 30 January, <https://emilyriederer.netlify.app/post/causal-design-patterns/>
- Sekhon, Jasjeet and Rocio Titiunik, 2016, ‘Understanding Regression Discontinuity Designs As Observational Studies’, *Observational Studies* 2 (2016) 174–182, <http://sekhon.berkeley.edu/papers/SekhonTitiunik2016-OS.pdf>.
- Wong, Jeffrey, and Colin McFarland, 2020, ‘Computational Causal Inference at Netflix’, *Netflix Technology Blog*, 11 Aug, <https://netflixtechblog.com/computational-causal-inference-at-netflix-293591691c62>.

### Required viewing

- Gelman, Andrew, 2020 ‘100 Stories of Causal Inference’, 4 August, <https://www.youtube.com/watch?v=jnI5KI843Lk>.
- King, Gary, 2020, ‘Research Designs’, Lectures on Quantitative Social Science Methods 1, <https://youtu.be/SBwPLwV0b7s>.
- Kuriwaki, Shiro, 2020, ‘Difference-in-Differences Estimation in R (parts 1 and 2)’, 18 April, <https://vimeo.com/409267138> and <https://vimeo.com/409267190>.
- Kuriwaki, Shiro, 2020, ‘Instrumental variables in R’, 11 April, <https://vimeo.com/406629459>.
- Kuriwaki, Shiro, 2020, ‘Regression Discontinuity in R (parts 1 and 2)’, 25 March, <https://vimeo.com/400826628> and <https://vimeo.com/400826660>.
- Oostrom, Tamar, 2021, ‘Funding of Clinical Trials and Reported Drug Efficacy’, 2 March, <https://youtu.be/DdnpWS9Km5U>.
- Riederer, Emily, 2021, ‘Observational Causal Inference’, Toronto Data Workshop, 15 February, <https://youtu.be/VP3BBZ7poc0>.

### Recommended reading

- Alexander, Monica, Polimis, Kivan, and Zagheni, Emilio, 2019, ‘The impact of Hurricane Maria on out-migration from Puerto Rico: Evidence from Facebook data’, *Population and Development Review*. (Example of using diff-in-diff to measure the effect of Hurricane Maria.)
- Alexander, Rohan, and Zachary Ward, 2018, ‘Age at arrival and assimilation during the age of mass migration’, *The Journal of Economic History*, 78, no. 3, 904–937. (Example where I used differences between brothers to estimate the effect of education.)
- Angrist, Joshua D., and Jörn-Steffen Pischke, 2008, *Mostly harmless econometrics: An empiricist’s companion*, Princeton University Press, Chapter 4.
- Angrist, Joshua D., and Jörn-Steffen Pischke, 2008, *Mostly harmless econometrics: An empiricist’s companion*, Princeton University Press, Chapter 6.
- Angrist, Joshua D., and Jörn-Steffen Pischke, 2008, *Mostly harmless econometrics: An empiricist’s companion*, Princeton University Press, Chapters 3.3.2 and 5.
- Austin, Peter C., 2011, ‘An Introduction to Propensity Score Methods for

Reducing the Effects of Confounding in Observational Studies', *Multivariate Behavioral Research*, vol. 46, no. 3, pp.399-424. (Broad overview of propensity score matching, with a nice discussion of the comparison to randomised controlled trials.)

- Baker, Andrew, 2019, 'Difference-in-Differences Methodology', 25 September, <https://andrewcbaker.netlify.app/2019/09/25/difference-in-differences-methodology/>.
- Coppock, Alenxader, and Donald P. Green, 2016, 'Is Voting Habit Forming? New Evidence from Experiments and Regression Discontinuities', *American Journal of Political Science*, Volume 60, Issue 4, pp. 1044-1062, available at: <https://onlinelibrary.wiley.com/doi/abs/10.1111/ajps.12210>. (Has code and data.)
- Cunningham, Scott, 'Causal Inference: The Mixtape', Chapter 'Instrumental variables', [http://www.scunning.com/causal\\_inference\\_norap.pdf](http://www.scunning.com/causal_inference_norap.pdf).
- Cunningham, Scott, 'Causal Inference: The Mixtape', chapter 'Regression discontinuity', [http://www.scunning.com/causal\\_inference\\_norap.pdf](http://www.scunning.com/causal_inference_norap.pdf).
- Cunningham, Scott, *Causal Inference: The Mixtape*, chapters 'Matching and subclassifications' and 'Differences-in-differences', [http://www.scunning.com/causal\\_inference\\_norap.pdf](http://www.scunning.com/causal_inference_norap.pdf). (Very well-written notes on diff-in-diff.)
- Dell, Melissa, Pablo Querubin, 2018, 'Nation Building Through Foreign Intervention: Evidence from Discontinuities in Military Strategies', *The Quarterly Journal of Economics*, Volume 133, Issue 2, pp. 701–764, <https://doi.org/10.1093/qje/qjx037>.
- Evans, David, 2013, 'Regression Discontinuity Porn', *World Bank Blogs*, 16 November, <https://blogs.worldbank.org/impactevaluations/regression-discontinuity-porn>.
- Gelman, Andrew, 2019, 'Another Regression Discontinuity Disaster and what can we learn from it', *Statistical Modeling, Causal Inference, and Social Science*, 25 June, <https://statmodeling.stat.columbia.edu/2019/06/25/another-regression-discontinuity-disaster-and-what-can-we-learn-from-it/>.
- Gelman, Andrew, and Guido Imbens, 2019, "Why high-order polynomials should not be used in regression discontinuity designs", *Journal of Business & Economic Statistics*, 37, pp. 447-456.
- Gelman, Andrew, and Jennifer Hill, 2007, *Data Analysis Using Regression and Multilevel/Hierarchical Models*, Chapter 10, pp. 207-215.
- Grogger, Jeffrey, Andreas Steinmayr, Joachim Winter, 2020, 'The Wage Penalty of Regional Accents', NBER Working Paper No. 26719.
- Harris, Rich, Mlacki Migliozi and Niraj Chokshi, '13,000 Missing Flights: The Global Consequences of the Coronavirus', *New York Times*, 21 February 2020. freely available here (if you make an account): <https://www.nytimes.com/interactive/2020/02/21/business/coronavirus-airline-travel.html>.
- Imai, Kosuke, 2017, Quantitative Social Science: An Introduction, Princeton University Press, Ch 2.5.

- Imbens, Guido W., and Thomas Lemieux, 2008, ‘Regression discontinuity designs: A guide to practice’, *Journal of Econometrics*, vol. 142, no. 2, pp. 615-635.
- King, Gary, and Richard Nielsen, 2019, ‘Why Propensity Scores Should Not Be Used for Matching’, *Political Analysis*. (Academic paper on the limits of propensity score matching. Propensity score matching was a big thing in the 90s but everyone knew about these weaknesses and so it died off. Lately, there has been a resurgence because of the CS/ML folks using it without thinking so King and Nielsen wrote a nice paper about the flaws. I mean, you can’t say you weren’t warned.)
- Myllyvirta, Lauri, 2020, ‘Analysis: Coronavirus has temporarily reduced China’s CO<sub>2</sub> emissions by a quarter’, *Carbon Brief*, 19 February, <https://www.carbonbrief.org/analysis-coronavirus-has-temporarily-reduced-chinas-co2-emissions-by-a-quarter>.
- Saeed, Sahar, Erica E. M. Moodie, Erin C. Strumpf, Marina B. Klein, 2019, ‘Evaluating the impact of health policies: using a difference-in-differences approach’, *International Journal of Public Health*, 64, pp. 637–642, <https://doi.org/10.1007/s00038-018-1195-2>.
- Taddy, Matt, 2019, *Business Data Science*, Chapter 5, pp. 146-162.
- Tang, John, 2015, ‘Pollution havens and the trade in toxic chemicals: evidence from U.S. trade flows’, *Ecological Economics*, vol. 112, pp. 150-160. (Example of using diff-in-diff to estimate pollution.)
- Travis, D.J., Carleton, A.M. and Lauritsen, R.G., 2004. ‘Regional variations in US diurnal temperature range for the 11–14 September 2001 aircraft groundings: Evidence of jet contrail influence on climate’, *Journal of climate*, 17(5), pp.1123-1134.
- Travis, David J., Andrew M. Carleton, and Ryan G. Lauritsen. “Contrails reduce daily temperature range.” *Nature*, 418, no. 6898 (2002): 601-601.
- Valencia Caicedo, Felipe. ‘The mission: Human capital transmission, economic persistence, and culture in South America.’ *The Quarterly Journal of Economics* 134.1 (2019): 507-556. (Data available at: Valencia Caicedo, Felipe, 2018, “Replication Data for: ‘The Mission: Human Capital Transmission, Economic Persistence, and Culture in South America’”, <https://doi.org/10.7910/DVN/ML1155>, Harvard Dataverse, V1.).
- Zinovyeva, Natalia and Maryna Tverdostup, 2019, ‘Why are women who earn slightly more than their husbands hard to find?’, 10 June, <https://blogs.lse.ac.uk/businessreview/2019/06/10/why-are-women-who-earn-slightly-more-than-their-husbands-hard-to-find/>.

### Key concepts/skills/etc

- Essential matching methods.
- Weaknesses of matching.
- Difference-in-differences.
- Identifying opportunities for instrumental variables.
- Implementing instrumental variables.

- Challenges to the validity of instrumental variables.
- Reading in foreign data.
- Difference in differences.
- Replicating work.
- Displaying multiple regression results.
- Discussing results.
- Generating simulated data.
- Understanding regression discontinuity and implementing it both manually and using packages.
- Appreciating the threats to the validity of regression discontinuity.

### Key libraries

- `broom`
- `tidyverse`
- `estimatr`
- `tidyverse`
- `haven`
- `huxtable`
- `scales`
- `tidyverse`
- `broom`
- `rdrobust`
- `tidyverse`

### Key functions/etc

- `tidy()`
- `lm()`
- `iv_robust()`
- `dollar_format()`
- `hux_reg()`
- `lm()`
- `mutate_at()`
- `read_dta()`
- `lm()`
- `tidy()`
- `rdrobust()()`

### Quiz

1. Sharla Gelfand has been ‘(s)haring two #rstats functions most days - one I know and love, and one that’s new to me!’. Please go to Sharla’s GitHub page: <https://github.com/sharlagelfand/twofunctionsmostdays>. Please find a package that she mentions that you have never used. Please find the relevant website for the pack-

- age. Please describe what the package does and a context in which it could be useful to you.
2. Sharla Gelfand has been ‘(s)haring two `#rstats` functions most days - one I know and love, and one that’s new to me!’. Please go to Sharla’s GitHub page: <https://github.com/sharlagelfand/twofunctionsmostdays>. Please find a function that she mentions that you have never used. Please look at the help file for that function. Please detail the arguments of the function, and a context in which it could be useful to you.
  3. What is propensity score matching? If you were matching people, then what are some of the features that you would like to match on? What sort of ethical questions does collecting and storing such information raise for you?
  4. Putting to one side, the ethical issues, what are some statistical weaknesses with propensity score matching?
  5. What is the key assumption when using diff-in-diff?
  6. Please read the fascinating article in The Markup about car insurance algorithms: <https://themarkup.org/allstates-algorithm/2020/02/25/car-insurance-suckers-list>. Please read the article and tell me what you think. You may wish to focus on ethical, legal, social, statistical, or other, aspects.
  7. Please go to the GitHub page related to the fascinating article in The Markup about car insurance algorithms: <https://github.com/the-markup/investigation-allstates-algorithm>. What is great about their work? What could be improved?
  8. What are the fundamental features of regression discontinuity design?
  9. What are the conditions that are needed in order for RDD to be able to be used?
  10. Can you think of a situation in your own life where RDD may be useful?
  11. What are some threats to the validity of RDD estimates?
  12. Please look at the `performance` package: <https://easystats.github.io/performance/index.html>. What are some features of this package that may be useful in your own work?
  13. What do you think about using COVID-19 in an RDD setting? Statistically? Ethically?
  14. Please read and reproduce the main findings from Eggers, Fowler, Hainmueller, Hall, Snyder, 2015.
  15. What is an instrumental variable?
  16. What are some circumstances in which instrumental variables might be useful?
  17. What conditions must instrumental variables satisfy?
  18. Who were some of the early instrumental variable authors?

19. Can you please think of and explain an application of instrumental variables in your own life?
20. What is the key assumption in difference-in-differences
  - a. Parallel trends.
  - b. Heteroscedasticity.
21. If you're using regression discontinuity, where are some aspects to be aware of and think really hard about (select all that apply)?
  - a. Is the cut-off free of manipulation?
  - b. Is the forcing function continuous?
  - c. To what extent is the functional form driving the estimate?
  - d. Would different fitted lines affect the results?
22. What is the main reason that [Oostrom \(2021\)](#) finds that the outcome of an RCT can depend on who is funding it (pick one)?
  - a. Publication bias
  - b. Explicit manipulation
  - c. Specialisation
  - d. Larger number of arms
23. What is the key coefficient of interest in Angelucci and Cagé, 2019 (pick one)?
  - a.  $\beta_0$
  - b.  $\beta_1$
  - c.  $\lambda$
  - d.  $\gamma$
24. The instrumental variable is (please pick all that apply):
  - a. Correlated with the treatment variable.
  - b. Not correlated with the outcome.
  - c. Heteroskedastic.
25. Who are the two candidates to have invented instrumental variables?
  - a. Sewall Wright
  - b. Philip G. Wright
  - c. Sewall Cunningham
  - d. Philip G. Cunningham
26. What are the two main assumptions of instrumental variables?
  - a. Exclusion Restriction.
  - b. Relevance.
  - c. Ignorability.
  - d. Randomization.
27. According to Meng, 2021, 'Data science can persuade via...' (pick all that apply):
  - a. the careful establishment of evidence from fair-minded and high-quality data collection
  - b. processing and analysis
  - c. the honest interpretation and communication of findings
  - d. large sample sizes

28. According to Reiderer, 2021, if I have ‘disjoint treated and untreated groups partitioned by a sharp cut-off’ then which method should I use to measure the local treatment effect at the juncture between groups (pick one)?
  - a. regression discontinuity
  - b. matching
  - c. difference-in-differences
  - d. event study methods
29. According to Reiderer, 2021, ‘Causal inference requires investment in’ (pick all that apply):
  - a. data management
  - b. domain knowledge
  - c. probabilistic reasoning
  - d. data science
30. I am an Australian 30-39 year old male living in Toronto with one child and a PhD. Which of the following do you think I would match most closely with and why (please explain in a paragraph or two)?
  - a. An Australian 30-39 year old male living in Toronto with one child and a bachelors degree
  - b. A Canadian 30-39 year old male living in Toronto with one child and a PhD
  - c. An Australian 30-39 year old male living in Ottawa with one child and a PhD
  - d. A Canadian 18-29 year old male living in Toronto with one child and a PhD
31. In your most disdainful tone (jokes, I love DAGs), what is a DAG (in your own words please)?
32. What is a confounder (please select one answer)?
  - a. A variable, z, that causes both x and y, where x also causes y.
  - b. A variable, z, that is caused by both x and y, where x also causes y.
  - c. A variable, z, that causes y and is caused by x, where x also causes y.
33. What is a mediator (please select one answer)?
  - a. A variable, z, that causes y and is caused by x, where x also causes y.
  - b. A variable, z, that causes both x and y, where x also causes y.
  - c. A variable, z, that is caused by both x and y, where x also causes y.
34. What is a collider (please select one answer)?
  - a. A variable, z, that causes both x and y, where x also causes y.
  - b. A variable, z, that causes y and is caused by x, where x also causes y.
  - c. A variable, z, that is caused by both x and y, where x also causes y.

35. Please talk through a brief example of when you may want to be very careful about checking for Simpson's paradox.
  36. Please talk through a brief example of when you may want to be very careful about checking for Berkson's paradox.
  37. According to McElreath (2020, 162) 'Regression will not sort it out. Regression is indeed an oracle, but a cruel one. It speaks in riddles and delights in punishing us for...' (please select one answer)?
    - a. overcomplicating models.
    - b. asking bad questions.
    - c. using bad data.
  38. Is a model that fits the small or large world more important to you, and why?
  39. In Kahneman et al. (2021) the authors, including the Nobel Prize winner Daniel Kahneman, say '... while correlation does not imply causation, causation does imply correlation. Where there is a causal link, we should find a correlation'. With reference to Cunningham (2021, Chapter 1), are they right or wrong, and why?
- 

## 16.1 Introduction

Life is grand when you can conduct experiments to be able to speak to causality. But what if you can only run the survey - you can't run an experiment? Here we begin our discussion of the circumstances and methods that would allow you to nonetheless speak to causality. We use (relatively) simple methods, in sophisticated, well-developed, ways (cf, much of what is done these days) and our applied statistics draw from a variety of social sciences including economics, and political science.

Following the publication of Dagan et al. (2021), one of the authors tweeted (slight edits for formatting):

---

We've just confirmed the effectiveness of the Pfizer-BioNTech vaccine outside of randomized trials. Yes, great news, but let's talk about methodological issues that arise when using observational data to estimate vaccine effectiveness.

A critical concern in observational studies of vaccine effectiveness is confounding: Suppose that people who get vaccinated have, on average, a lower risk of infection/disease than those who don't get vaccinated. Then, even if the vaccine were useless, it'd look beneficial.

To adjust for confounding: We start by identifying potential confounders. For example: Age (vaccination campaigns prioritize older people and older people are more likely to develop severe disease). Then we choose a valid adjustment method. In our paper, we matched on age. After age adjustment, how do we know if there is residual confounding? Here is one way to go about that: We know from the previous randomized trial that the vaccine has no effect in the first few days. So we check whether matching on age suffices to replicate that finding. No, it doesn't. After matching on age (and sex), the curves of infection start to diverge from day 0, which indicates that the vaccinated had a lower risk of infection than the unvaccinated. Conclusion: adjustment for age and sex is insufficient.

We learned that we had to match on other COVID-19 risk factors, e.g., location, comorbidities, healthcare use... And we could do so with high-quality data from the Clalit Research Institute, part of a health services organization that covers >50% of the Israeli population. As an example, a vaccinated 76 year-old Arab male from a specific neighborhood who received 4 influenza vaccines in the last 5 years and had 2 comorbidities was matched with an unvaccinated Arab male from the same neighborhood, aged 76-77, with 3-4 influenza vaccines and 2 comorbidities. After matching on all those risk factors, the curves of infection start to diverge after day ~12, as expected if the vaccinated and the unvaccinated had a comparable risk of infection. Using this “negative control”, we provide evidence against large residual confounding.

This is a good illustration of how randomized trials and observational studies complement each other for better and more efficient #causalinfERENCE. First, a randomized trial is conducted to estimate the effectiveness of the vaccine to prevent symptomatic infection, but... the trial's estimates for severe disease and specific age groups are imprecise. Second, an observational analysis emulates a #targettrial (an order of magnitude greater) and confirms the vaccine's effectiveness on severe disease and in different age groups. However... the observational study needs the trial's findings as a benchmark to guide the data analysis and strengthen the quality of the causal inference.

Randomized trials & Observational studies working together. The best of both worlds. Let's keep doing it after the pandemic. What a luxury having been able to think about these issues with my colleagues Noa Dagan, Noam Barda, Marc Lipsitch, Ben Reis, and Ran D. Balicer. We hope that our experience is

helpful for researchers around the world who use observational data to estimate vaccine effectiveness.

Miguel Hernán, 24 February 2021<sup>1</sup>.

---

This is what this chapter is about. How we can nonetheless be comfortable making causal statements, even when we can't run A/B tests or RCTs. Indeed, in what circumstances may we actually prefer to not run those or to run observational-based approaches in addition to them. We cover three of the major methods that are in popular use these days: difference-in-differences; regression discontinuity; and instrumental variables.

---

## 16.2 DAGs and trying not to be tricked by the data

### 16.2.1 DAGs and confounding

When we are discussing causality it can help to be very specific about what we mean. It's easy to get caught up in the data that it tricks you. It's important to think really hard. One framework to help with this that has become popular recently is the use of directed acyclic graph (DAG), which is essentially just a fancy name for a flow diagram. A DAG involves drawing arrows between your variables indicating the relationship between them. We will use the `DiagrammeR` package to draw them (Iannone, 2020), because that provides quite a lot of control (Figure 16.1). However it can be a little finicky and if you're just looking to do something really quickly then the `ggdag` package can be useful (Barrett, 2021b). The code to draw these DAGs draws heavily on Igelström (2020).

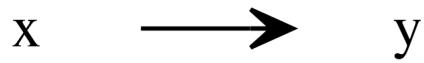
```
library(DiagrammeR)
DiagrammeR::grViz("
digraph {
 graph [ranksep = 0.2]
 node [shape = plaintext]
 x
 y
 edge [minlen = 2, arrowhead = vee]
 x->y
 { rank = same; x; y }
```

---

<sup>1</sup>[https://twitter.com/\\_miguelhernan/status/1364700315044438023?s=21](https://twitter.com/_miguelhernan/status/1364700315044438023?s=21)

```
}
```

```
")
```



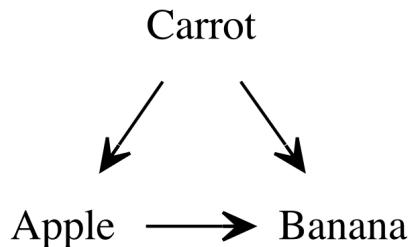
**FIGURE 16.1:** Using a DAG to illustrate perceived relationships

In this example, we claim that  $x$  causes  $y$ . We could build another where the situation is less clear. I find all the  $x$  and  $y$  very confusing, so will change to fruits (Figure 16.2).

```
DiagrammeR::grViz("
digraph {
 graph [ranksep = 0.2]
 node [shape = plaintext]
 Apple
 Banana
 Carrot
 edge [minlen = 2, arrowhead = vee]
 Apple->Banana
 Carrot->Apple
 Carrot->Banana
 { rank = same; Apple; Banana }
}
")
```

In this case we again think *apple* causes *banana*. But it's also clear that *carrot* causes *banana*, and *carrot* also causes *apple*. That relationship is a 'backdoor path', and would create spurious correlation in our analysis. We may think that changes in *apple* are causing changes in *banana*, but it's actually that *carrot* is changing them both and hence that variable is called a 'confounder'.

There is an excellent discussion by Hernan and Robins (2020, p. 83):



**FIGURE 16.2:** Carrot as a confounder

Suppose an investigator conducted an observational study to answer the causal question “does one’s looking up to the sky make other pedestrians look up too?” She found an association between a first pedestrian’s looking up and a second one’s looking up. However, she also found that pedestrians tend to look up when they hear a thunderous noise above. Thus it was unclear what was making the second pedestrian look up, the first pedestrian’s looking up or the thunderous noise? She concluded the effect of one’s looking up was confounded by the presence of a thunderous noise.

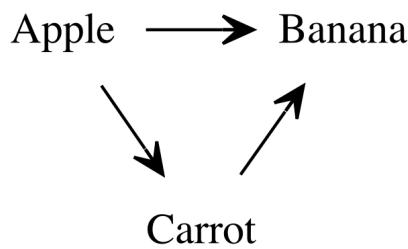
In randomized experiments treatment is assigned by the flip of a coin, but in observational studies treatment (e.g., a person’s looking up) may be determined by many factors (e.g., a thunderous noise). If those factors affect the risk of developing the outcome (e.g., another person’s looking up), then the effects of those factors become entangled with the effect of treatment. We then say that there is confounding, which is just a form of lack of exchangeability between the treated and the untreated. Confounding is often viewed as the main shortcoming of observational studies. In the presence of confounding, the old adage “association is not causation” holds even if the study population is arbitrarily large.

If we were interested in causal effects we would need to adjust for *carrot*, or

*thunder* and one way is to include it in the regression. However, the validity of this requires a number of assumptions. In particular, Gelman and Hill (2007, p. 169) warns us our estimate will only correspond to the average causal effect in the sample if: 1) we include ‘all confounding covariates’; and 2) ‘the model is correct’. Putting to one side the second requirement, and focusing only on the first, if we don’t observe a confounder, then we can’t adjust for it. This is where the role of domain experts, experience, and theory, really add a lot to an analysis.

We might have a similar situation, again we think that *apple* causes *banana*, but this time *apple* also causes *carrot*, which itself causes *banana* (Figure 16.3).

```
DiagrammeR::grViz("
digraph {
 graph [ranksep = 0.2]
 node [shape = plaintext]
 Apple
 Banana
 Carrot
 edge [minlen = 2, arrowhead = vee]
 Apple->Banana
 Apple->Carrot
 Carrot->Banana
 { rank = same; Apple; Banana }
}
")
```

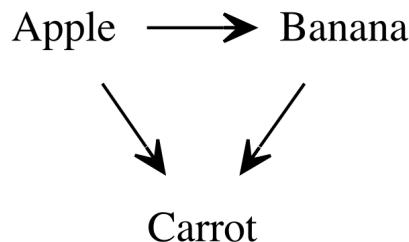


**FIGURE 16.3:** Carrot as a mediator

In this case, *carrot* is called a ‘mediator’ and we’d not like to adjust for it, because that would affect our estimate of the effect of *apple* on *banana*.

Finally, we might have yet another similar situation, where we again think that *apple* causes *banana*, but this time both *apple* and *banana* cause *carrot* (Figure 16.4).

```
DiagrammeR::grViz("
digraph {
 graph [ranksep = 0.2]
 node [shape = plaintext]
 Apple
 Banana
 Carrot
 edge [minlen = 2, arrowhead = vee]
 Apple->Banana
 Apple->Carrot
 Banana->Carrot
 { rank = same; Apple; Banana }
}
")
```



**FIGURE 16.4:** Carrot as a collider

In this case, *carrot* is called a ‘collider’ and if we were to condition on it we would create a misleading relationship.

I’ve been circling around this point for a while, but it’s time to address it. You will create your DAG - there is nothing that will create it for you. That means that you need to think really carefully about the situation. Because it’s one thing to see something in the DAG and then do something about it. But it’s another to not know that it’s there. McElreath (2020, p. 180) describes these as haunted DAGs.

DAGs have become fashionable. They are of course helpful, but they are just

a tool to help you think really deeply about your situation. As [McElreath \(2020, p. 162\)](#) says ‘Regression will not sort it out. Regression is indeed an oracle, but a cruel one. It speaks in riddles and delights in punishing us for asking bad questions.’. The same is true of DAGs, and all the methods that we cover in these notes.

### 16.2.2 Selection and measurement bias

Selection bias occurs when the outcomes are dependent on ‘the process by which individuals are selected into the analysis’ ([Hernan and Robins, 2020, p. 99](#)). We are not going to be able to see this from a DAG (I mean one could draw a DAG that shows it, but you need to know about it in order to draw that DAG is what I mean), or many default diagnostics. One way to go about things is A/A testing in an experimental settings, or comparing the sample with some more general characteristics, for instance age-group, gender, and education. But the fundamental point, as Dr Jill Sheppard says, is that ‘people who respond to surveys are weird’, and this generalises to whatever method you’re using to gather your data. Using Facebook ads? People who click on Facebook ads are weird. Going door knocking? People who answer their door are weird. Call people on the phone? Literally who answers their phone anymore.

There is a pernicious aspect of selection bias, which is that it pervades every aspect of your analysis. Even a sample that starts off as perfectly representative, may become selected over time. For instance, the survey panels used for political polling need to be updated from time to time because the folks who don’t get anything out of it stop responding - but those people may still vote and so need to be polled.

Another bias to be very aware of is measurement bias, which when ‘the association between treatment and outcome is weakened or strengthened as a result of the process by which the study data are measured’ ([Hernan and Robins, 2020, p. 113](#)). It is implicit in the definition by [Hernan and Robins \(2020\)](#), but it is important here that this be systematic. For instance, if I ask people in person what their income is then that is likely to get different answers than if I ask over the phone or via an online form.

### 16.2.3 Two common paradoxes

There are two situations where data can trick you that are so common that I’d like to explicitly go through them. These are Simpson’s paradox, and Berkson’s paradox. Keep these situations in the back of your mind at all times when dealing with data.

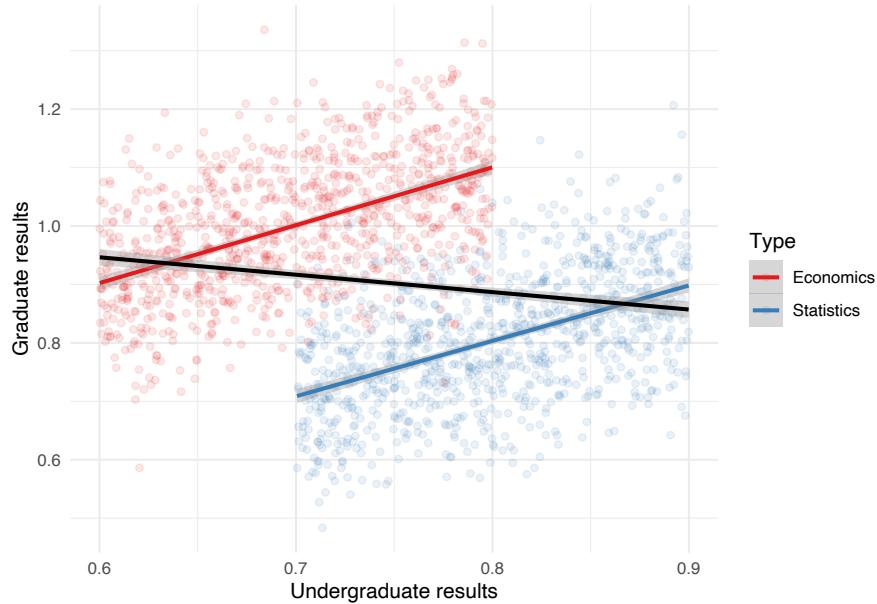
Simpson’s paradox occurs when we estimate some relationship for subsets of our data, but a different relationship when we consider the entire dataset

(Simpson, 1951). For instance, it may be that there is a positive relationship between undergraduate grades and performance in graduate school in both statistics and economics when considering each department individually. But if undergraduate grades tended to be higher in statistics than economics while graduate school performance tended to be opposite, we may actually find a negative relationship between undergraduate grades and performance in graduate school.

To see this let's simulate some data.

```
set.seed(853)
number_in_each <- 1000
statistics <- tibble(undergrad = runif(n = number_in_each, min = 0.7, max = 0.9),
 noise = rnorm(n = number_in_each, 0, sd = 0.1),
 grad = undergrad + noise,
 type = "Statistics")
economics <- tibble(undergrad = runif(n = number_in_each, min = 0.6, max = 0.8),
 noise = rnorm(n = number_in_each, 0, sd = 0.1),
 grad = undergrad + noise + 0.3,
 type = "Economics")
both = rbind(statistics, economics)

both %>%
 ggplot(aes(x = undergrad, y = grad)) +
 geom_point(aes(color = type), alpha = 0.1) +
 geom_smooth(aes(color = type), method = 'lm', formula = 'y ~ x') +
 geom_smooth(method = 'lm', formula = 'y ~ x', color = 'black') +
 labs(x = "Undergraduate results",
 y = "Graduate results",
 color = "Type") +
 theme_minimal() +
 scale_color_brewer(palette = "Set1")
```



Berkson's paradox occurs when we estimate some relationship based on the dataset that we have, but because the dataset is selected the relationship is different in a more general dataset (Berkson, 1946). For instance, if we have a dataset of professional cyclists then we would find there is no relationship between their VO<sub>2</sub> max and their chance of winning a bike race. But if we had a dataset of the general population then we would find an enormous relationship between their VO<sub>2</sub> max and their chance of winning a bike race. The professional dataset has just been so selected that the relationship disappears - you can't become a professional unless you have a high VO<sub>2</sub> max.

To see this let's simulate some data.

```
set.seed(853)
number_of_pros <- 100
number_of_public <- 1000
professionals <-
 tibble(VO2 = runif(n = number_of_pros, min = 0.7, max = 0.9),
 chance_of_winning = runif(n = number_of_pros, min = 0.7, max = 0.9),
 type = "Professionals")
general_public <-
 tibble(VO2 = runif(n = number_of_public, min = 0.6, max = 0.8),
 noise = rnorm(n = number_of_public, 0, sd = 0.03),
 chance_of_winning = VO2 + noise + 0.1,
 type = "Public") %>%
 select(-noise)
```

```
both = rbind(professionals, general_public)

both %>%
 ggplot(aes(x = VO2, y = chance_of_winning)) +
 geom_point(aes(color = type), alpha = 0.1) +
 geom_smooth(aes(color = type), method = 'lm', formula = 'y ~ x') +
 geom_smooth(method = 'lm', formula = 'y ~ x', color = 'black') +
 labs(x = "VO2 max",
 y = "Chance of winning a bike race",
 color = "Type") +
 theme_minimal() +
 scale_color_brewer(palette = "Set1")
```



## 16.3 Difference in differences

### 16.3.1 Matching and difference-in-differences

#### 16.3.1.1 Introduction

The ideal situation of being able to conduct an experiment is rarely possible in a data science setting. Can we really reasonably expect that Netflix would

allow us to change prices. And even if they did once, would they let us do it again, and again, and again? Further, rarely can we explicitly create treatment and control groups. Finally, experiments are really expensive and potentially unethical. Instead, we need to make do with what we have. Rather than our counterfactual coming to us through randomisation, and hence us knowing that the two are the same but for the treatment, we try to identify groups that were similar before the treatment, and hence any differences can be attributed to the treatment. In practice, we tend to even have differences between our two groups before we treat. Provided those pre-treatment differences satisfy some assumptions (basically that they were consistent, and we expect that consistency to continue in the absence of the treatment) – the ‘parallel trends’ assumption – then we can look to any difference in the differences as the effect of the treatment. One of the lovely aspects of difference in differences analysis is that we can do it using fairly straight-forward quantitative methods - linear regression with a dummy variable is all that is needed to do a convincing job.

#### 16.3.1.2 Motivation

Consider us wanting to know the effect of a new tennis racket on serve speed. One way to test this would be to measure the difference between Roger Federer’s serve speed without the tennis racket and mine with the tennis racket. Sure, we’d find a difference but how do we know how much to attribute to the tennis racket? Another way would be to consider the difference between my serve speed without the tennis racket and my serve speed with the tennis racket. But what if serves were just getting faster naturally over time? Instead, let’s combine the two to look at the difference in the differences!

In this world we measure Federer’s serve and compare it to my serve without the new racket. We then measure Federer’s serve again and measure my serve with the new racket. That difference in the differences would then be our estimate of the effect of the new racket.

What sorts of assumptions jump out at you that we are going to have to make in order for this analysis to be appropriate?

- 1) Is there something else that may have affected only me, and not Roger that could affect my serve speed? Probably.
- 2) Is it likely that Roger Federer and I have the same trajectory of serve speed improvement? Probably not. This is the ‘parallel trends’ assumption, and it dominates any discussion of difference in differences analysis. Finally, is it likely that the variance of our serve speeds is the same? Probably not.

Why might this be powerful? We don’t need the treatment and control group to be the same before the treatment. We just need to have a good idea of how they differ.

### 16.3.1.3 Simulated example

Let's generate some data.

```
library(broom)
library(tidyverse)

set.seed(853)

diff_in_diff_example_data <- tibble(person = rep(c(1:1000), times = 2),
 time = c(rep(0, times = 1000), rep(1, times = 1000)),
 treatment_group = rep(sample(x = 0:1, size = 1000, replace = TRUE), times = 2))

We want to make the outcome slightly more likely if they were treated than if not.
diff_in_diff_example_data <-
 diff_in_diff_example_data %>%
 rowwise() %>%
 mutate(serve_speed = case_when(
 time == 0 & treatment_group == 0 ~ rnorm(n = 1, mean = 5, sd = 1),
 time == 1 & treatment_group == 0 ~ rnorm(n = 1, mean = 6, sd = 1),
 time == 0 & treatment_group == 1 ~ rnorm(n = 1, mean = 8, sd = 1),
 time == 1 & treatment_group == 1 ~ rnorm(n = 1, mean = 14, sd = 1),
)
)

head(diff_in_diff_example_data)
#> # A tibble: 6 x 4
#> # Rowwise:
#> person time treatment_group serve_speed
#> <int> <dbl> <int> <dbl>
#> 1 1 0 0 4.43
#> 2 2 0 1 6.96
#> 3 3 0 1 7.77
#> 4 4 0 0 5.31
#> 5 5 0 0 4.09
#> 6 6 0 0 4.85
```

Let's make a graph.

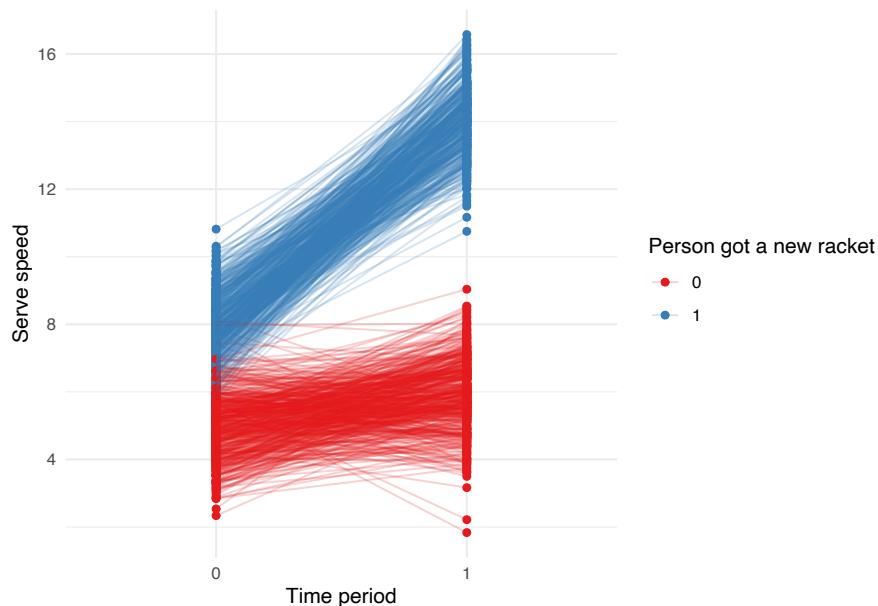
```
diff_in_diff_example_data$treatment_group <- as.factor(diff_in_diff_example_data$treatment_group)
diff_in_diff_example_data$time <- as.factor(diff_in_diff_example_data$time)

diff_in_diff_example_data %>%
 ggplot(aes(x = time,
 y = serve_speed,
```

```

color = treatment_group)) +
geom_point() +
geom_line(aes(group = person), alpha = 0.2) +
theme_minimal() +
labs(x = "Time period",
y = "Serve speed",
color = "Person got a new racket") +
scale_color_brewer(palette = "Set1")

```



As it is a simple example, we could do this manually, by getting the average difference of the differences.

```

average_differences <-
diff_in_diff_example_data %>%
pivot_wider(names_from = time,
values_from = serve_speed,
names_prefix = "time_") %>%
mutate(difference = time_1 - time_0) %>%
group_by(treatment_group) %>%
summarise(average_difference = mean(difference))

average_differences$average_difference[2] - average_differences$average_difference[1]
#> [1] 5.058414

```

Let's use OLS to do the same analysis. The general regression equation is:

$$Y_{i,t} = \beta_0 + \beta_1 \text{Treatment group dummy}_i + \beta_2 \text{Time dummy}_t + \beta_3 (\text{Treatment group dummy} \times \text{Time dummy})_{i,t} + \epsilon_{i,t}$$

If we use \* in the regression then it automatically includes the separate aspects as well as their interaction. It's the estimate of  $\beta_3$  which is of interest.

```
diff_in_diff_example_regression <- lm(serve_speed ~ treatment_group*time,
 data = diff_in_diff_example_data)

tidy(diff_in_diff_example_regression)
#> # A tibble: 4 x 5
#> term estimate std.error statistic p.value
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 (Intercept) 4.97 0.0428 116. 0
#> 2 treatment_group1 3.03 0.0622 48.7 0
#> 3 time1 1.01 0.0605 16.6 2.97e-58
#> 4 treatment_group1:time1 5.06 0.0880 57.5 0
```

Fortunately, our estimates are the same!

#### 16.3.1.4 Assumptions

If we want to use difference in differences, then we need to satisfy the assumptions. There were three that were touched on earlier, but here I want to focus on one: the ‘parallel trends’ assumption. The parallel trends assumption haunts everything to do with diff-in-diff analysis because we can never prove it, we can just be convinced of it.

To see why we can never prove it, consider an example in which we want to know the effect of a new stadium on a professional sports team’s wins/loses. To do this we consider two teams: the Warriors and the Raptors. The Warriors changed stadiums at the start of the 2019-20 season (the Raptors did not), so we will consider four time periods: the 2016-17 season, 2017-18 season, 2018-19 season, and finally we will compare the performance with the one after they moved, so in the 2019-20 season. The Raptors here act as our counterfactual. This means that we assume the relationship between the Warriors and the Raptors, in the absence of a new stadium, would have continued to change in a consistent way. But we can never know that for certain. We have to present sufficient evidence to assuage any concerns that a reader may have.

For a variety of reasons, it is worth having tougher than normal requirements around the evidence it would take to convince you of an effect.

There are four main ‘threats to validity’ when you are using difference in differences and you should address all of these (Cunningham, 2020, pp. 272–277):

1. Non-parallel trends. The treatment and control groups may be based on differences. As such it can be difficult to convincingly argue for parallel trends. In this case, maybe try to find another factor to consider in your model that may adjust for some of that. This may require difference in difference in differences (in the earlier example, perhaps could add in the San Francisco 49ers as they are in the same broad geographic area as the Warriors). Or maybe rethink your analysis to see if you can make a different control group. Adding additional earlier time periods may help but may introduce more issues (see third point).
2. Compositional differences. This is a concern when working with repeated cross-sections. What if the composition of those cross-sections change? For instance, if we work at Tik Tok or some other app that is rapidly growing and want to look at the effect of some change. In our initial cross-section, we may have mostly young people, but in a subsequent cross-section, we may have more older people as the demographics of the app usage change. Hence our results may just be an age-effect, not an effect of the change that we are interested in.
3. Long-term effects vs. reliability. As we discussed in the last chapter, there is a trade-off between the length of the analysis that we run. As we run the analysis for longer there is more opportunity for other factors to affect the results. There is also increased chance for someone who was not treated to be treated. But, on the other hand, it can be difficult to convincingly argue that short-term results will continue in the long-term.
4. Functional form dependence. This is less of an issue when the outcomes are similar, but if they are different then functional form may be responsible for some aspects of the results.

#### 16.3.1.5 Matching

*This section draws on material from Gelman and Hill, 2007, pp. 207-212.*

Difference in differences is a powerful analysis framework. After I learnt about it I began to see opportunities to implement it everywhere. But it can be tough to identify appropriate treatment and control groups. In Alexander and Ward, 2018, we compare migrant brothers - one of whom had most of their education in a different country, and the other who had most of their education in the US. Is this really the best match?

We may be able to match based on observable variables. For instance, age-group or education. At two different times we compare smoking rates in 18-year-olds in one city with smoking rates in 18-year-olds in another city. That is fine, but it is fairly coarse. We know that there are differences between 18-year-olds, even in terms of the variables that we commonly observe, say

gender and education. One way to deal with this may be to create sub-groups: 18-year-old males with a high school education, etc. But the sample sizes are likely to quickly become small. How do we deal with continuous variables? And also, is the difference between an 18-year-old and a 19-year-old really so different? Shouldn't we also compare with them?

One way to proceed is to consider a nearest neighbour approach. But there is limited concern for uncertainty in this approach. There is also an issue if you have a large number of variables because you end up with a high-dimension graph. This leads us to propensity score matching.

Propensity score matching involves assigning some probability to each observation. We construct that probability based on the observation's values for the independent variables, at their values before the treatment. That probability is our best guess at the probability of the observation being treated, regardless of whether it was treated or not. For instance, if 18-year-old males were treated but 19-year-old males were not, then as there is not much difference between 18-year-old males and 19-year-old males our assigned probability would be fairly similar. We can then compare the outcomes of observations with similar propensity scores.

One advantage of propensity score matching is that it allows us to easily consider many independent variables at once, and it can be constructed using logistic regression.

Let's generate some data to illustrate propensity score matching. Let's pretend that we work for Amazon. We are going to treat some individuals with free-shipping to see what happens to their average purchase.

```
library(tidyverse)

sample_size <- 10000
set.seed(853)

amazon_purchase_data <-
 tibble(
 unique_person_id = c(1:sample_size),
 age = runif(n = sample_size,
 min = 18,
 max = 100),
 city = sample(
 x = c("Toronto", "Montreal", "Calgary"),
 size = sample_size,
 replace = TRUE
),
```

```

gender = sample(
 x = c("Female", "Male", "Other/decline"),
 size = sample_size,
 replace = TRUE,
 prob = c(0.49, 0.47, 0.02)
),
income = rlnorm(n = sample_size,
 meanlog = 0.5,
 sdlog = 1)
)

```

Now we need to add some probability of being treated with free shipping, which depends on our variables. Younger, higher-income, male and in Toronto all make it slightly more likely.

```

amazon_purchase_data <-
 amazon_purchase_data %>%
 mutate(age_num = case_when(
 age < 30 ~ 3,
 age < 50 ~ 2,
 age < 70 ~ 1,
 TRUE ~ 0),
 city_num = case_when(
 city == "Toronto" ~ 3,
 city == "Montreal" ~ 2,
 city == "Calgary" ~ 1,
 TRUE ~ 0),
 gender_num = case_when(
 gender == "Male" ~ 3,
 gender == "Female" ~ 2,
 gender == "Other/decline" ~ 1,
 TRUE ~ 0),
 income_num = case_when(
 income > 3 ~ 3,
 income > 2 ~ 2,
 income > 1 ~ 1,
 TRUE ~ 0)
) %>%
rowwise() %>%
 mutate(sum_num = sum(age_num, city_num, gender_num, income_num),
 softmax_prob = exp(sum_num)/exp(12),
 free_shipping = sample(
 x = c(0:1),

```

```

 size = 1,
 replace = TRUE,
 prob = c(1-softmax_prob, softmax_prob)
)
) %>%
ungroup()

amazon_purchase_data <-
 amazon_purchase_data %>%
 dplyr::select(-age_num, -city_num, -gender_num, -income_num, -sum_num, -softmax_prob)

```

Finally, we need to have some measure of a person's average spend. We want those with free shipping to be slightly higher than those without.

```

amazon_purchase_data <-
 amazon_purchase_data %>%
 mutate(mean_spend = if_else(free_shipping == 1, 60, 50)) %>%
 rowwise() %>%
 mutate(average_spend = rnorm(1, mean_spend, sd = 5)
) %>%
ungroup() %>%
dplyr::select(-mean_spend)

Fix the class on some
amazon_purchase_data <-
 amazon_purchase_data %>%
 mutate_at(vars(city, gender, free_shipping), ~as.factor(.)) ## Change some to factors

```

```

table(amazon_purchase_data$free_shipping)
#>
#> 0 1
#> 9629 371

head(amazon_purchase_data)
#> # A tibble: 6 x 7
#> unique_person_id age city gender income free_shipping
#> <int> <dbl> <fct> <fct> <dbl> <fct>
#> 1 1 47.5 Calgary Female 1.72 0
#> 2 2 27.8 Montreal Male 1.54 0
#> 3 3 57.7 Toronto Female 3.16 0
#> 4 4 43.9 Toronto Male 0.636 0
#> 5 5 21.1 Toronto Female 1.43 0

```

```
#> 6 6 51.1 Calgary Male 1.18 0
#> # ... with 1 more variable: average_spend <dbl>
```

Now we construct a logistic regression model that ‘explains’ whether a person was treated as a function of the variables that we think explain it.

```
propensity_score <- glm(free_shipping ~ age + city + gender + income,
 family = binomial,
 data = amazon_purchase_data)
```

We will now add our forecast to our dataset.

```
amazon_purchase_data <-
 augment(propensity_score,
 data = amazon_purchase_data,
 type.predict = "response") %>%
 dplyr::select(-.resid, -.std.resid, -.hat, -.sigma, -.cooksdi)
```

Now we use our forecast to create matches. There are a variety of ways to do this. In a moment I’ll step through some code that does it all at once, but as this is a worked example and we only have a small number of possibilities, we can just do it manually.

For every person who was actually treated (given free shipping) we want the untreated person who was considered as similar to them (based on propensity score) as possible.

```
amazon_purchase_data <-
 amazon_purchase_data %>%
 arrange(.fitted, free_shipping)
```

Here we’re going to use a matching function from the `arm` package. This finds which is the closest of the ones that were not treated, to each one that was treated.

```
amazon_purchase_data$treated <- if_else(amazon_purchase_data$free_shipping == 0, 0, 1)
amazon_purchase_data$treated <- as.integer(amazon_purchase_data$treated)

matches <- arm::matching(z = amazon_purchase_data$treated, score = amazon_purchase_data$.fitted)

amazon_purchase_data <- cbind(amazon_purchase_data, matches)
```

Now we reduce the dataset to just those that are matched. We had 371 treated, so we expect a dataset of 742 observations.

```
amazon_purchase_data_matched <-
 amazon_purchase_data %>%
 filter(match.ind != 0) %>%
 dplyr::select(-match.ind, -pairs, -treated)

head(amazon_purchase_data_matched)
#> unique_person_id age city gender income
#> 1 5710 81.15636 Montreal Female 0.67505625
#> 2 9458 97.04859 Montreal Female 9.49752179
#> 3 6428 83.21262 Calgary Male 0.05851482
#> 4 2022 98.97504 Montreal Male 1.66683768
#> 5 9824 64.61936 Calgary Female 3.35263989
#> 6 1272 97.09546 Toronto Female 0.71813784
#> free_shipping average_spend .fitted cnts
#> 1 0 47.36258 0.001375987 1
#> 2 1 61.15317 0.001376161 1
#> 3 0 49.90080 0.001560150 1
#> 4 1 57.75673 0.001560418 1
#> 5 1 64.69709 0.002207195 1
#> 6 0 56.64754 0.002207514 1
```

Finally, we can examine the ‘effect’ of being treated on average spend in the ‘usual’ way.

```
propensity_score_regression <- lm(average_spend ~ age + city + gender + income + free_shipping,
 data = amazon_purchase_data_matched)
```

```
huxtable::huxreg(propensity_score_regression)
```

(1)
(Intercept)
49.694 ***
(0.809)
age
0.005
(0.011)
cityMontreal

0.169  
(0.734)  
cityToronto  
0.652  
(0.623)  
genderMale  
-0.968 \*  
(0.422)  
genderOther/decline  
-1.973  
(2.621)  
income  
0.009  
(0.021)  
free\_shipping1  
10.488 \*\*\*  
(0.380)  
N  
742  
R2  
0.513  
logLik  
-2267.486  
AIC  
4552.971

\*\*\* p < 0.001; \*\* p < 0.01; \* p < 0.05.

I cover propensity score matching here because it is widely used. Hence, you need to know how to use it. People would think it's weird if you didn't, in the same way that we have to cover ANOVA people would think it's weird if we had an entire experimental design course and didn't cover it even though there are more modern ways of looking at differences between two means. But

## 16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974

at the same time you need to know that there are flaws with propensity score matching. I will now discuss some of them.

1. Matching. Propensity score matching cannot match on unobserved variables. This may be fine in a class-room setting, but in more realistic settings it will likely cause issues.
  2. Modelling. The results tend to be specific to the model that is used. King and Nielsen, 2019, discuss this thoroughly.
  3. Statistically. We are using the data twice.
- 

## 16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974

### 16.4.1 Introduction

In this case study we introduce Angelucci and Cagé, 2019, and replicate its main findings. Angelucci and Cagé, 2019, is a paper in which difference in differences is used to examine the effect of the reduction in advertising revenues on newspapers' content and prices. They create a dataset of 'French newspapers between 1960 and 1974'. They 'perform a difference-in-differences analysis' and exploit 'the introduction of advertising on television' as this change 'affected national newspapers more severely than local ones'. They 'find robust evidence of a decrease in the amount of journalistic-intensive content produced and the subscription price.'

In order to conduct this analysis we will use the dataset that they provide alongside their paper. This dataset is available at: <https://www.openicpsr.org/openicpsr/project/116438/version/V1/view>. It is available for you to download after registration. As their dataset is in Stata data format, we will use the haven package to read it in (Wickham and Miller, 2019).

```
library(here)
library(haven)
library(huxtable)
#>
#> Attaching package: 'huxtable'
#> The following objects are masked from 'package:ggdag':
#>
#> label, label<-
#> The following object is masked from 'package:dplyr':
#>
#> add_rownames
```

```
#> The following object is masked from 'package:ggplot2':
#>
#> theme_grey
library(scales)
#>
#> Attaching package: 'scales'
#> The following object is masked from 'package:huxtable':
#>
#> number_format
#> The following object is masked from 'package:purrr':
#>
#> discard
#> The following object is masked from 'package:readr':
#>
#> col_factor
library(tidyverse)
```

### 16.4.2 Background

Newspapers are in trouble. We can probably all think of a local newspaper that has closed recently because of pressure brought on by the internet. But this issue isn't new. When television started, there were similar concerns. In this paper, Angelucci and Cagé use the introduction of television advertising in France, announced in 1967, to examine the effect of decreased advertising revenue on newspapers.

The reason this is important is because it allows us to disentangle a few competing effects. For instance, are newspapers becoming redundant because they can no longer charge high prices for their ads or because consumers prefer to get their news in other ways? Are fewer journalists needed because smartphones and other technology mean they can be more productive? Angelucci and Cagé look at advertising revenue and a few other features, when a new advertising platform arrives, in this case television advertising.

### 16.4.3 Data

---

(The) dataset contains annual data on local and national newspapers between 1960 and 1974, as well as detailed information on television content. In 1967, the French government announced it would relax long-standing regulations that prohibited television advertising. We provide evidence that this reform can be

#### 16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974

plausibly interpreted as an exogenous and negative shock to the advertising side of the newspaper industry... [I]t is likely that the introduction of television advertising constituted a direct shock to the advertising side of the newspaper industry and only an indirect shock to the reader side... (O)ur empirical setting constitutes a unique opportunity to isolate the consequences of a decrease in newspapers' advertising revenues on their choices regarding the size of their newsroom, the amount of information to produce, and the prices they charge to both sides of the market.

---

The authors' argue that national newspapers were affected by the television advertising change, but local newspapers were not. So the national newspapers are the treatment group and the local newspapers are the control group.

The dataset can be read in using `read_dta()`, which is a function within the `haven` package for reading in Stata dta files. This is equivalent to `read_csv()`.

```
newspapers <- read_dta(here::here("inputs/data/116438-V1/data/dta/Angelucci_Cage_AEJMicro_dataset.dta"))

dim(newspapers)
#> [1] 1196 52
```

There are 1,196 observations in the dataset and 52 variables. The authors are interested in the 1960-1974 time period which has around 100 newspapers. There are 14 national newspapers at the beginning of the period and 12 at the end.

We just want to replicate their main results, so we don't need all their variables. As such we will just `select()` the ones that we are interested in and change the `class()` where needed.

```
newspapers <-
 newspapers %>%
 dplyr::select(year, id_news, after_national, local, national, ## Diff in diff variables
 ra_cst, qtotal, ads_p4_cst, ads_s, ## Advertising side dependents
 ps_cst, po_cst, qtotal, qs_s, rs_cst) %>% #Reader side dependents
 mutate(ra_cst_div_qtotal = ra_cst / qtotal) %>% ## An advertising side dependents needs to be built
 mutate_at(vars(id_news, after_national, local, national), ~as.factor(.)) %>% ## Change some to factors
 mutate(year = as.integer(year))
```

We can now have a look at the main variables of interest for both national (Figure 16.5) and local daily newspapers (Figure 16.6).

Source: Angelucci and Cagé, 2019, p. 333.

	Mean	Median	SD	Min	Max	Observations
<i>Prices</i>						
Unit buyer price	3.6	3.5	1.3	2.4	9.3	152
Subscription price per issue	2.8	2.7	0.7	1.9	5.6	148
Display ad rate (listed price)	121.1	114.5	81.0	17.5	274.2	121
<i>Revenues and journalists</i>						
Total revenues (million €)	425	271	403	19	1,482	162
Revenues from advertising (million €)	228	103	258	7	864	161
Revenues from sales (million €)	199	145	181	12	657	162
Share of advertising in total revenues (percent)	47.4	51.1	21.3	8.0	81.0	162
Number of journalists	117	85	81	21	326	158
<i>Circulation</i>						
Total circulation	295,210	181,574	292,838	16,112	1,143,676	162
Share of subscribers (percent)	25.6	18.5	26.3	0.7	92.3	163
<i>Content</i>						
Number of pages	19	17	7	8	38	138
Newshole (nonadvertising space)	13	13	4	6	25	138
Advertising space	5	4	4	0	16	138

*Notes:* The table gives summary statistics. The time period is 1960–1974. Variables are values for newspapers. The observations are at the newspaper/year level. Unit price, subscription price per issue, and list price are in constant (2014) euros. Revenues and costs are in million constant (2014) euros.

**FIGURE 16.5:** Angelucci and Cagé, 2019, summary statistics: national daily newspapers

	Mean	Median	SD	Min	Max	Observations
<i>Prices</i>						
Unit buyer price	3.2	3.3	0.8	0.8	5.7	911
Subscription price per issue	2.8	2.9	0.7	0.7	4.7	896
Display ad rate (listed price)	80.3	57.7	72.6	3.8	327.2	688
<i>Revenues and journalists</i>						
Total revenues (million €)	146	65	176	1	1,026	888
Revenues from advertising (million €)	67	30	79	1	416	891
Revenues from sales (million €)	79	36	102	0	751	884
Share of advertising in total revenues (percent)	46.5	45.9	8.3	7.1	70.4	878
Number of journalists	53	27	58	1	297	907
<i>Circulation</i>						
Total circulation	101,487	50,586	119,774	1,480	654,992	908
Share of subscribers (percent)	27.5	23.3	22.0	1.0	100.1	909
<i>Content</i>						
Number of pages	15	15	6	2	66	908
Newshole (nonadvertising space)	12	12	4	2	34	908
Advertising space	3	2	3	0	32	908

*Notes:* The table gives summary statistics. The time period is 1960–1974. Variables are values for newspapers. The observations are at the newspaper/year level. Unit price, subscription price per issue, and list price are in constant (2014) euros. Revenues and costs are in million constant (2014) euros.

**FIGURE 16.6:** Angelucci and Cagé, 2019, summary statistics: local daily newspapers

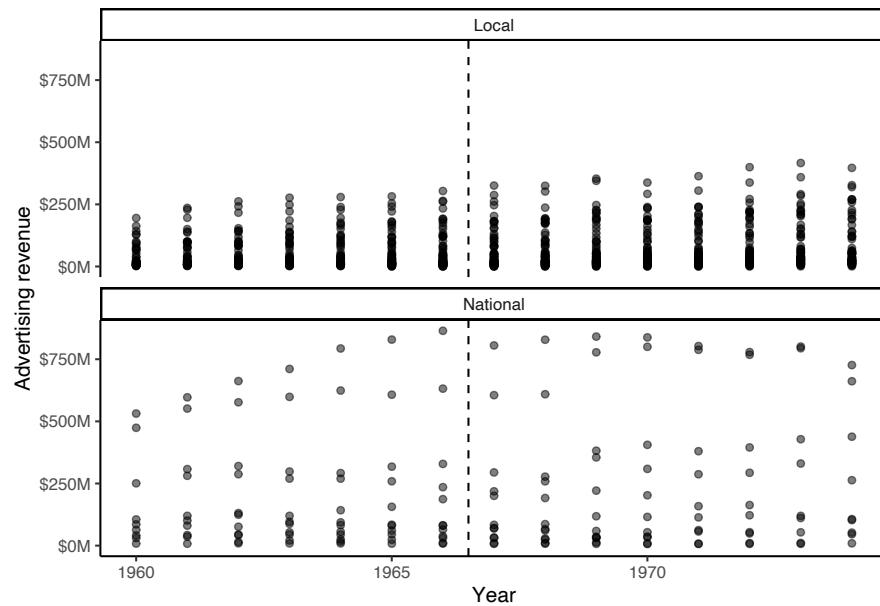
## 16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974

Source: Angelucci and Cagé, 2019, p. 334.

Please read this section of their paper to see how they describe their dataset.

We are interested in the change from 1967 onward.

```
newspapers %>%
 mutate(type = if_else(local == 1, "Local", "National")) %>%
 ggplot(aes(x = year, y = ra_cst)) +
 geom_point(alpha = 0.5) +
 scale_y_continuous(labels = dollar_format(prefix="$", suffix = "M", scale = 0.000001)) +
 labs(x = "Year",
 y = "Advertising revenue") +
 facet_wrap(vars(type),
 nrow = 2) +
 theme_classic() +
 geom_vline(xintercept = 1966.5, linetype = "dashed")
```



### 16.4.4 Model

The model that we are interested in estimating is:

$$\ln(y_{n,t}) = \beta_0 + \beta_1(\text{National dummy} \times 1967 \text{ onward dummy}) + \lambda_n + \gamma_y + \epsilon.$$

The  $\lambda_n$  is a fixed effect for each newspaper, and the  $\gamma_y$  is a fixed effect for each year. We just use regular linear regression, with a few different dependent variables. It is the  $\beta_1$  coefficient that we are interested in.

### 16.4.5 Results

We can run the models using `lm()`.

```
Advertising side
ad_revenue <- lm(log(ra_cst) ~ after_national + id_news + year, data = newspapers)
ad_revenue_div_circulation <- lm(log(ra_cst_div_qtotal) ~ after_national + id_news + year, data = newspapers)
ad_price <- lm(log(ads_p4_cst) ~ after_national + id_news + year, data = newspapers)
ad_space <- lm(log(ads_s) ~ after_national + id_news + year, data = newspapers)

Consumer side
subscription_price <- lm(log(ps_cst) ~ after_national + id_news + year, data = newspapers)
unit_price <- lm(log(po_cst) ~ after_national + id_news + year, data = newspapers)
circulation <- lm(log(qtotal) ~ after_national + id_news + year, data = newspapers)
share_of_sub <- lm(log(qs_s) ~ after_national + id_news + year, data = newspapers)
revenue_from_sales <- lm(log(rs_cst) ~ after_national + id_news + year, data = newspapers)
```

Looking at the advertising-side variables.

```
omit_me <- c("(Intercept)", "id_news3", "id_news6", "id_news7", "id_news13",
 "id_news16", "id_news25", "id_news28", "id_news34", "id_news38",
 "id_news44", "id_news48", "id_news51", "id_news53", "id_news54",
 "id_news57", "id_news60", "id_news62", "id_news66", "id_news67",
 "id_news70", "id_news71", "id_news72", "id_news80", "id_news82",
 "id_news88", "id_news95", "id_news97", "id_news98", "id_news103",
 "id_news105", "id_news106", "id_news118", "id_news119", "id_news127",
 "id_news136", "id_news138", "id_news148", "id_news151", "id_news153",
 "id_news154", "id_news157", "id_news158", "id_news161", "id_news163",
 "id_news167", "id_news169", "id_news179", "id_news184", "id_news185",
 "id_news187", "id_news196", "id_news206", "id_news210", "id_news212",
 "id_news213", "id_news224", "id_news225", "id_news234", "id_news236",
 "id_news245", "id_news247", "id_news310", "id_news452", "id_news467",
 "id_news469", "id_news480", "id_news20040", "id_news20345",
 "id_news20346", "id_news20347", "id_news20352", "id_news20354",
 "id_news21006", "id_news21025", "id_news21173", "id_news21176",
 "id_news33718", "id_news34689", "id_news73")

huxreg("Ad. rev." = ad_revenue,
 "Ad rev. div. circ." = ad_revenue_div_circulation,
 "Ad price" = ad_price,
 "Ad space" = ad_space,
 omit_coefs = omit_me,
 number_format = 2
)
```

*16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974* 561

Ad. rev.

Ad rev. div. circ.

Ad price

Ad space

after\_national1

-0.23 \*\*\*

-0.15 \*\*\*

-0.31 \*\*\*

0.01

(0.03)

(0.03)

(0.07)

(0.05)

year

0.05 \*\*\*

0.04 \*\*\*

0.04 \*\*\*

0.02 \*\*\*

(0.00)

(0.00)

(0.00)

(0.00)

N

1052

1048

809

1046

R2

0.99

0.90

0.89

0.72

logLik

345.34

449.52

-277.71

-164.01

AIC

-526.68

-735.05

705.43

478.02

\*\*\* p &lt; 0.001; \*\* p &lt; 0.01; \* p &lt; 0.05.

Similarly, we can look at the reader-side variables.

```
omit_me <- c("(Intercept)", "id_news3", "id_news6", "id_news7", "id_news13",
 "id_news16", "id_news25", "id_news28", "id_news34", "id_news38",
 "id_news44", "id_news48", "id_news51", "id_news53", "id_news54",
 "id_news57", "id_news60", "id_news62", "id_news66", "id_news67",
 "id_news70", "id_news71", "id_news72", "id_news80", "id_news82",
 "id_news88", "id_news95", "id_news97", "id_news98", "id_news103",
 "id_news105", "id_news106", "id_news118", "id_news119", "id_news127",
 "id_news136", "id_news138", "id_news148", "id_news151", "id_news153",
 "id_news154", "id_news157", "id_news158", "id_news161", "id_news163",
 "id_news167", "id_news169", "id_news179", "id_news184", "id_news185",
 "id_news187", "id_news196", "id_news206", "id_news210", "id_news212",
 "id_news213", "id_news224", "id_news225", "id_news234", "id_news236",
 "id_news245", "id_news247", "id_news310", "id_news452", "id_news467",
 "id_news469", "id_news480", "id_news20040", "id_news20345",
 "id_news20346", "id_news20347", "id_news20352", "id_news20354",
 "id_news21006", "id_news21025", "id_news21173", "id_news21176",
 "id_news33718", "id_news34689", "id_news73")
```

```
huxreg("Subscription price" = subscription_price,
 "Unit price" = unit_price,
 "Circulation" = circulation,
 "Share of sub" = share_of_sub,
 "Revenue from sales" = revenue_from_sales,
```

#### 16.4 Case study - Lower advertising revenue reduced French newspaper prices between 1960 and 1974563

```
 omit_coefs = omit_me,
 number_format = 2
)
```

Subscription price

Unit price

Circulation

Share of sub

Revenue from sales

after\_national1

-0.04 \*

0.06 \*\*

-0.06 \*\*

0.19 \*\*\*

-0.06 \*

(0.02)

(0.02)

(0.02)

(0.03)

(0.03)

year

0.05 \*\*\*

0.05 \*\*\*

0.01 \*\*\*

-0.01 \*\*\*

0.05 \*\*\*

(0.00)

(0.00)

(0.00)

(0.00)

(0.00)

N  
1044  
1063  
1070  
1072  
1046  
R2  
0.88  
0.87  
0.99  
0.97  
0.99  
logLik  
882.14  
907.28  
759.57  
321.91  
451.11  
AIC  
-1600.28  
-1650.57  
-1355.15  
-477.81  
-738.22

\*\*\* p < 0.001; \*\* p < 0.01; \* p < 0.05.

#### 16.4.6 Other points

- We certainly find that in many cases there appears to be a difference from 1967 onward.
- In general, we are able to obtain results that are similar to Angelucci and Cagé, 2019. If we spent more time, we could probably replicate their findings perfectly. Isn't this great! What else could do?

## *16.5 Case study - Funding of Clinical Trials and Reported Drug Efficacy*

- Parallel trends: Notice the wonderful way in which they test the ‘parallel trends’ assumption on pp. 350-351.
- Discussion: Look at their wonderful discussion (pp. 353-358) of interpretation, external validity, and robustness.

→

---

## **16.5 Case study - Funding of Clinical Trials and Reported Drug Efficacy**

Oostrom (2021) looks at clinical trials of drugs. These days, of course, we all know a lot more than we may have ever wished to, about clinical trials. But the one thing that we (think) we know is that they are, well, clinical. By that I mean, that it doesn’t matter who does the actual trial, the outcome would be the same. Oostrom (2021) says this isn’t true.

By way of background, clinical trials are needed before a drug can be approved. Oostrom (2021) finds that when pharmaceutical firms sponsor a clinical trial, ‘a drug appears 0.15 standard deviations more effective when the trial is sponsored by that drug’s manufacturer, compared with the same drug in the same trial without the drug manufacturer’s involvement.’ She does this by exploiting the fact that often ‘the exact same sets of drugs are often compared in different randomized control trials conducted by parties with different financial interests.’

The main finding is (Oostrom, 2021, p. 2):

---

Utilizing dozens of drug combinations across hundreds of clinical trials, I estimate that a drug appears 36 percent more effective (0.15 standard deviations off of a base of 0.42) when the trial is sponsored by that drug’s manufacturing or marketing firm, compared with the same drug, evaluated against the same comparators, but without the drug manufacturer’s involvement. As in the medical literature, I measure efficacy, in the case of antidepressants, as the share of patients that respond to medication or, in the case of schizophrenia, as the average decline in symptoms.

---

Why might this happen? Oostrom (2021) looks at a variety of different options,

grouped into those that happen before the trial and those that happen after the trial ‘publication bias’. She finds that ‘publication bias can explain as much as half of this sponsorship effect. Incorporating data on unpublished clinical trials, I find sponsored trials are less likely to publish non-positive results for their drugs.’

Oostrom (2021) focuses on antidepressant and antipsychotic drugs and this allows her to obtain a dataset of trials. An ‘arm’ of a trial refers to ‘the unit at which randomization occurs. Arms are often unique drugs but occasionally refer to unique drug and dosage combinations.’ (Oostrom, 2021, p. 9).

Summary statistics are provided in a summary table (Figure 16.7) (this approach is common in economics, but not a great idea because it hides the distribution of the data - better to plot the raw data.)

Table 1: Summary Statistics: Full and Variation Samples

	Full Sample				Sample with Variation within:				Drug Pairs			
	Drug Sets		Drug Pairs									
	Mean	Median	Std Dev.	% Missing	Mean	Median	Std Dev.	% Missing	Mean	Median	Std Dev.	% Missing
Year	2001	2001	9	14	1999	1999	8	8	2000	2000	8	12
Year relative to approval	10.1	8.0	10.6	39	11.4	8.0	11.4	24	11.8	9.0	10.8	34
Share:												
Sponsored	0.49	0.00	0.50	0	0.52	1.00	0.50	0	0.42	0.00	0.49	0
Sponsored, no COI	0.43	0.00	0.50	0	0.41	0.00	0.49	0	0.34	0.00	0.47	0
Antidepressant	0.77	1.00	0.42	0	0.81	1.00	0.39	0	0.81	1.00	0.39	0
Registered	0.13	0.00	0.33	0	0.05	0.00	0.22	0	0.09	0.00	0.29	0
Trial design:												
# of patients	100	89	84	0	88	70	98	0	94	78	90	0
Length (weeks)	8.7	8.0	7.5	0	8.5	6.0	6.4	0	9.1	8.0	8.1	0
Standard outcome	0.90	1.00	0.30	0	0.90	1.00	0.31	0	0.90	1.00	0.30	0
Baseline severity	-0.0	-0.0	1.0	7	0.0	-0.0	1.0	6	-0.1	-0.1	1.0	8
Dosage (mg)	68	25	102	26	59	20	91	17	58	20	85	26
% Dropout	29	27	15	10	29	27	15	11	30	28	15	12
Mean age	42	41	9	18	44	41	11	17	43	41	9	19
% Female	52	58	20	44	52	55	20	54	52	57	20	50
Total arms	1,412				499				900			
Total trials	586				230				400			

Notes: This table presents summary statistics at the trial arm level. Summary statistics are shown for the full sample, the subsample with variation in sponsorship within drug sets, and the subsample with variation in sponsorship within drug pairs. Details for each outcome are listed in Section 2.3.4.

**FIGURE 16.7:** Summary statistics from Ooostrom

The model is:

$$y_{ij} = \alpha + \beta \text{Sponsor}_{ij} + X_{ij}\gamma + G_{d(i),s(j)} + \epsilon_{ij}$$

where  $y_{ij}$  is the efficacy for arm  $i$  in trial  $j$ . The main coefficient of interest is  $\beta$  which is based on whether Sponsor $_{ij}$ . The outcome is relative to the placebo arm in that trial, or the least effective arm.

‘Table 3.3’ from the paper is actually the reason that I included this as a case student. If this sounds odd to you then you’ve not had to read millions of papers that are unclear about their results. ‘Table 3.3’ (republished here as Figure 16.8) is beautiful and I’ll allow it to speak for itself.

### 3.3 Difference in Difference Framework

The empirical framework in this paper can be succinctly summarized in Table 3. This contains all antidepressant drug sets that compare one active drug to a placebo and have variation in sponsorship (the “Active vs. Placebo” row in Table 2).<sup>25</sup> Each row is a unique drug set and, in my initial empirical specification, each drug in each row would receive its own fixed effect.

Table 3: Difference in Difference: Active versus Placebo Antidepressants

All Drug Sets	Sponsored				Not Sponsored				DD	
	Share Respond			# Trials	Share Respond			# Trials		
	Drug	Placebo	Diff		Drug	Placebo	Diff			
All Drug Sets	0.491	0.303	0.188	59	0.441	0.301	0.140	8	0.048	
Paroxetine	0.469	0.320	0.149	32	0.250	0.226	0.024	1	0.126	
Sertraline	0.453	0.360	0.093	12	0.476	0.433	0.042	2	0.051	
Citalopram	0.513	0.399	0.114	8	0.303	0.209	0.095	1	0.019	
Trazodone	0.458	0.158	0.300	6	0.568	0.353	0.215	1	0.085	
Amitriptyline	0.564	0.278	0.286	1	0.607	0.282	0.325	3	-0.039	

Notes: This table presents the difference-in-difference estimate of the sponsorship effect for “Active vs. Placebo” drug sets. The first set of columns compares the share of patients that respond to treatment when the drug is sponsored; the next set compare these results when the drug is not sponsored. The difference between the share of patients that respond to a given drug and the share that respond to the placebo group is given in the column labeled “Diff” for “Difference.” The last column reports the difference in difference (DD) is analogous to the sponsorship effect in equation 1.

## FIGURE 16.8: Results

The paper is available here: <https://www.tamaroostrom.com/research> and I’d recommend a brief read.

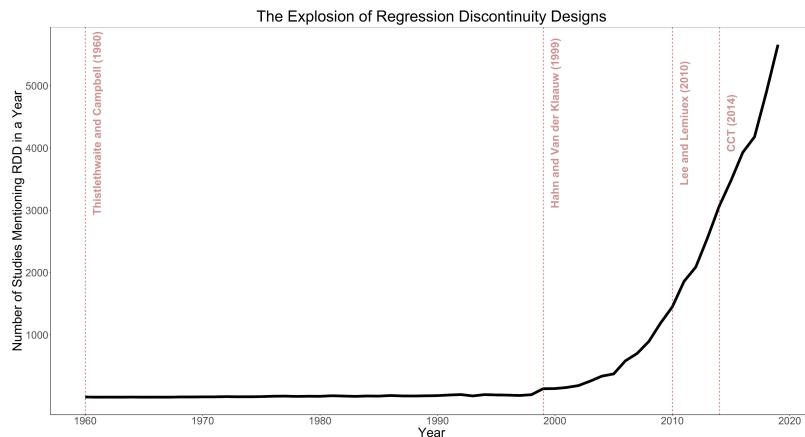
## 16.6 Regression discontinuity design

### 16.6.1 Introduction

Regression discontinuity design (RDD) is a popular way to get causality when there is some continuous variable with cut-offs that determine treatment. Is there a difference between a student who gets 79 per cent and a student who

gets 80 per cent? Probably not much, but one gets an A-, while the other gets a B+, and seeing that on a transcript could affect who gets a job which could affect income. In this case the percentage is a ‘forcing variable’ and the cut-off for an A- is a ‘threshold’. As the treatment is determined by the forcing variable all you need to do is to control for that variable. And, these seemingly arbitrary cut-offs can be seen all the time. Hence, there has been an ‘explosion’ in the use of regression discontinuity design (Figure 16.9).

Please note that I’ve followed the terminology of Taddy, 2019. Gelman and Hill, 2007, and others use slightly different terminology. For instance, Cunningham refers to the forcing function as the running variable. It doesn’t matter what you use so long as you are consistent. If you have a terminology that you are familiar with then please feel free to use it, and to share it with me!



**FIGURE 16.9:** The explosion of regression discontinuity designs in recent years.

Source: John Holbein, 13 February 2020<sup>2</sup>.

The key assumptions are:

1. The cut-off is ‘known, precise and free of manipulation’ (Cunningham, 2020, p. 163).
2. The forcing function should be continuous because this means we can say that people on either side of the threshold are the same, other than happening to just fall on either side of the threshold.

### 16.6.2 Simulated example

Let’s generate some data.

<sup>2</sup><https://twitter.com/JohnHolbein1/status/1228050675378069504>

```
library(broom)
library(tidyverse)

set.seed(853)

number_of_observation <- 1000

rdd_example_data <- tibble(person = c(1:number_of_observation),
 grade = runif(number_of_observation, min = 78, max = 82),
 income = rnorm(number_of_observation, 10, 1)
)

We want to make income more likely to be higher if they are have a grade over 80
rdd_example_data <-
 rdd_example_data %>%
 mutate(income = if_else(grade > 80, income + 2, income))

head(rdd_example_data)
```

person

grade

income

1

79.4

9.43

2

78.5

9.69

3

79.9

10.8

4

79.3

9.34

5

78.1

10.7

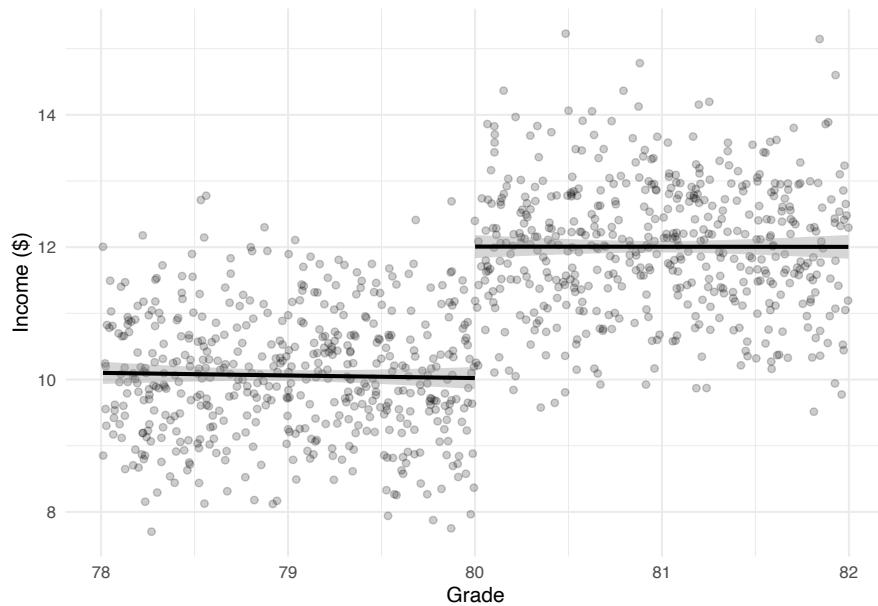
6

79.6

9.83

Let's make a graph.

```
rdd_example_data %>%
 ggplot(aes(x = grade,
 y = income)) +
 geom_point(alpha = 0.2) +
 geom_smooth(data = rdd_example_data %>% filter(grade < 80),
 method='lm',
 color = "black") +
 geom_smooth(data = rdd_example_data %>% filter(grade >= 80),
 method='lm',
 color = "black") +
 theme_minimal() +
 labs(x = "Grade",
 y = "Income ($)")
#> `geom_smooth()` using formula 'y ~ x'
#> `geom_smooth()` using formula 'y ~ x'
```



We can use a dummy variable with linear regression to estimate the effect (we're hoping that it's 2 because that is what we imposed.)

```
rdd_example_data <-
 rdd_example_data %>%
 mutate(grade_80_and_over = if_else(grade < 80, 0, 1))

lm(income ~ grade + grade_80_and_over, data = rdd_example_data) %>%
 tidy()
```

term	estimate
estimate	std.error
std.error	statistic
statistic	p.value
p.value	(Intercept)
(Intercept)	11.7
11.7	4.24
4.24	2.76
2.76	0.00585
0.00585	grade
grade	-0.021
-0.021	0.0537
0.0537	-0.391
-0.391	0.696
0.696	grade_80_and_over
grade_80_and_over	1.99
1.99	0.123
0.123	16.2
16.2	1.34e-52

There are various caveats to this estimate that we'll get into later, but the essentials are here.

The other great thing about regression discontinuity is that it can almost be as good as an RCT. For instance, (and I thank John Holbein for the pointer)

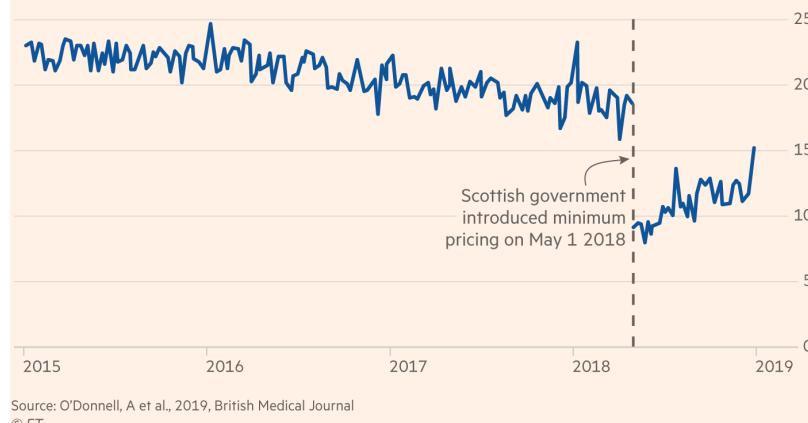
Bloom et al. (2020) compare randomized trials with RDDs and find that the RCTs compare favourably.

#### 16.6.2.1 Different slopes

Figure 16.10 shows an example with different slopes.

**After Scotland introduced minimum unit pricing for alcohol, consumption fell overnight**

Grams of alcohol purchased per adult per household, Scotland minus England



Source: O'Donnell, A et al., 2019, British Medical Journal  
© FT

**FIGURE 16.10:** Effect of minimum unit pricing for alcohol in Scotland.

Source: John Burn-Murdoch, 7 February 2020<sup>3</sup>.

#### 16.6.3 Overlap

In the randomised control trial and A/B testing section, because of randomised assignment of the treatment, we imposed that the control and treatment groups were the same but for the treatment. We moved to difference-in-differences, and we assumed that there was a common trend between the treated and control groups. We allowed that the groups could be different, but that we could ‘difference out’ their differences. Finally, we considered matching, and we said that even if the control and treatment groups seemed quite different we were able to match those who were treated with a group that were similar to them in all ways, apart from the fact that they were not treated.

In regression discontinuity we consider a slightly different setting - the two groups are completely different in terms of the forcing variable - they are on either side of the threshold. So there is no overlap at all. But we know

<sup>3</sup><https://twitter.com/jburnmurdoch/status/1225773931342303233>

the threshold and believe that those on either side are essentially matched. Let's consider the 2019 NBA Eastern Conference Semifinals - Toronto and the Philadelphia. Game 1: Raptors win 108-95; Game 2: 76ers win 94-89; Game 3: 76ers win 116-95; Game 4: Raptors win 101-96; Game 5: Raptors win 125-89; Game 6: 76ers win 112-101; and finally, Game 7: Raptors win 92-90, because of a ball that went in after bouncing on the rim four times. Was there really that much difference between the teams (Figure 16.11)?



**FIGURE 16.11:** It took four bounces to go in, so how different were the teams...?

Source: Stan Behal / Postmedia Network<sup>4</sup>.

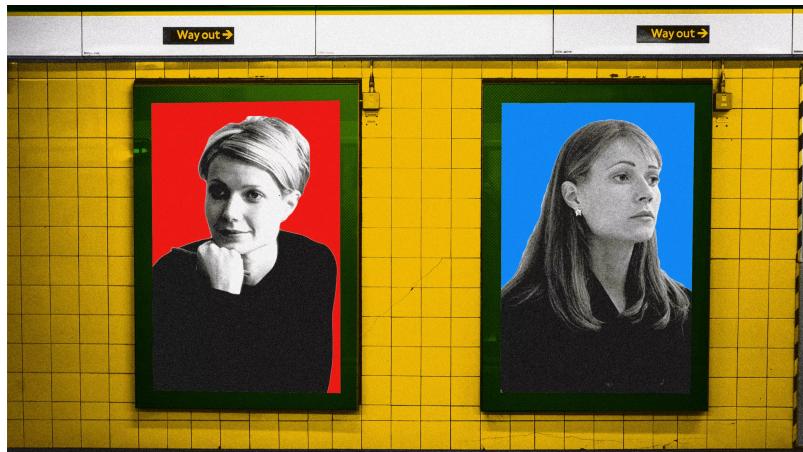
#### 16.6.4 Examples

As with difference-in-differences, after I learnt about it, I began to see opportunities to implement it everywhere. Frankly, I find it a lot easier to think of legit-

---

<sup>4</sup><https://nationalpost.com/sports/kawhi-leonard-miracle-shot-toronto-raptors-game-7-scott-stinson-kawhi-leonards-miracle-shot-sends-toronto-raptors-into-game-7-elation>

imate examples of using regression discontinuity than difference-in-differences. But, at the risk of mentioning yet another movie from the 1990s that none of you have seen, when I think of RDD, my first thought is often of Sliding Doors (Figure 16.12).



**FIGURE 16.12:** Nobody expects the Spanish Inquisition.

Source: Mlotek, Haley, 2018, ‘The Almosts and What-ifs of ’Sliding Doors”, *The Ringer*, 24 April, freely available at: <https://www.theringer.com/movies/2018/4/24/17261506/sliding-doors-20th-anniversary>.

Not only did the movie have a great soundtrack and help propel Gwyneth Paltrow to super-stardom, but it features an iconic moment in which Paltrow’s character, Helen, arrives at a tube station at which point the movie splits into two. In one version she just makes the train, and arrives home to find her boyfriend cheating on her; and in another she just misses the train and doesn’t find out about the boyfriend.

I’d say, spoiler alert, but the movie was released in 1998, so... Of course, that ‘threshold’ turns out to be important. In the world in which she gets the train she leaves the boyfriend, cuts her hair, and changes everything about her life. In the world in which she misses the train she doesn’t. At least initially. But, and I can’t say this any better than Ashley Fetter:

---

At the end of Sliding Doors, the “bad” version of Helen’s life elides right into the “good” version; even in the “bad” version, the philandering !@#\$%^& boyfriend eventually gets found out and dumped, the true love eventually gets met-cute, and the

MVP friend comes through. According to the Sliding Doors philosophy, in other words, even when our lives take fluky, chaotic detours, ultimately good-hearted people find each other, and the bad boyfriends and home-wreckers of the world get their comeuppance. There's no freak turn of events that allows the cheating boyfriend to just keep cheating, or the well-meaning, morally upright soulmates to just keep floating around in the universe unacquainted.

Fetters, Ashley, 2018, 'I Think About This a Lot: The Sliding Doors in Sliding Doors', *The Cut*, 9 April, freely available at: <https://www.thecut.com/2018/04/i-think-about-this-a-lot-the-sliding-doors-in-sliding-doors.html>.

---

I'm getting off-track here, but the point is, not only does it seem as though we have a 'threshold', but it seems as though there's continuity!

Let's see some more legitimate implementations of regression discontinuity. (And thank you to Ryan Edwards<sup>5</sup> for pointing me to these.)

#### 16.6.4.1 Elections

Elections are a common area of application for regression discontinuity because if the election is close then arguably there's not much difference between the candidates. There are plenty of examples of regression discontinuity in an elections setting, but one recent one is George, Siddharth Eapen, 2019, 'Like Father, Like Son? The Effect of Political Dynasties on Economic Development', freely available at: [https://www.dropbox.com/s/orvh3n03wd9ybl/sid\\_JMP\\_dynasties\\_latestdraft.pdf?dl=0](https://www.dropbox.com/s/orvh3n03wd9ybl/sid_JMP_dynasties_latestdraft.pdf?dl=0).

In this paper George is interested in political dynasties. But is the child of a politician more likely to be elected because they are the child of a politician, or because they happen to also be similarly skilled at politics? Regression discontinuity can help because in a close election, we can look at differences between places where someone narrowly won with where a similar someone narrowly lost.

In the George, 2019, case he examines:

---

descendant effects using a close elections regression discontinuity (RD) design. We focus on close races between dynastic de-

<sup>5</sup><http://www.ryanbedwards.com/>

scendants (i.e. direct relatives of former officeholders) and non-dynasts, and we compare places where a descendant narrowly won to those where a descendant narrowly lost. In these elections, descendants and non-dynasts have similar demographic and political characteristics, and win in similar places and at similar rates. Nevertheless, we find negative economic effects when a descendant narrowly wins. Villages represented by a descendant have lower asset ownership and public good provision after an electoral term: households are less likely to live in a brick house and to own basic amenities like a refrigerator, mobile phone, or vehicle. Moreover, voters assess descendants to perform worse in office. An additional standard deviation of exposure to descendants lowers a village's wealth rank by 12pp.

The model that George, 2019, estimates is (p. 19):  
 $y_i = \alpha_{\text{district}} + \beta \times \text{Years descendant rule}_i + f(\text{Descendant margin}) + \gamma X_i + \epsilon_{i,t}$ .

In this model,  $y_i$  is various development outcomes in village  $i$ ; Years descendant rule $_i$  is the number of years a dynastic descendant has represented village  $i$  in the national or state parliament; Descendant margin is the vote share difference between the dynastic descendant and non-dynast; and  $\gamma X_i$  is a vector of village-level adjustments.

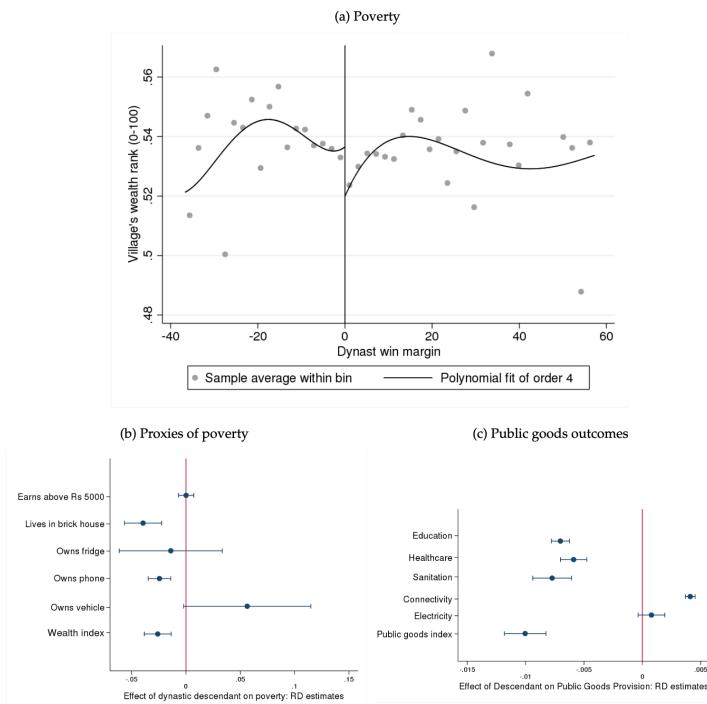
George, 2019, then conducts a whole bunch of tests of the validity of the regression discontinuity design (p. 19). These are critical in order for the results to be believed. There are a lot of different results but one is shown in Figure 16.13.

#### 16.6.4.2 Economic development

One of the issues with considering economic development is that a place typically is either subject to some treatment or not. However, sometimes regression discontinuity allows us to compare areas that were just barely treated with those that were just barely not.

One recent paper that does this Esteban Mendez-Chacon and Diana Van Patten, 2020, 'Multinationals, monopsony and local development: Evidence from the United Fruit Company' available here: <https://www.dianavanpatten.com/>. They are interested in the effect of the United Fruit Company (UFCo), which was given land in Costa Rica between 1889 and 1984. They were given roughly 4 per cent of the national territory or around 4500 acres. The key is that this land assignment was redrawn in 1904 based on a river and hence the re-assignment was essentially random with regard to determinants of growth to that point. They compare areas that were assigned to UFCo with those that were not. They find:

Figure 2: Descendant effects identified using close elections RD design



This figure presents results from a regression discontinuity design specification. We restrict attention to elections where the top two candidates are a dynastic descendant and a non-dynast (or vice versa). The running variable is *Descendant win margin*, which is defined as the vote share of the dynastic descendant minus the vote share of non-dynast. The outcomes in Panels A and C are measures of poverty. The key outcome variable is *Wealth index*, which is based on data from the Socioeconomic and Caste Census. The underlying variables used to construct the index are: (i) *Earns > Rs 5k*: the share of households earning over Rs 5000 ( $\approx$ US\$70) per month, (ii) *Brick house*: the share of households in a village living in a structure with at least a brick or concrete wall and roof, (iii) *Fridge, Phone and Vehicle*: the share of households in a village who own these assets. We take the principal component of these variables and assign villages a percentile rank (between 0 and 1) based on their index score. Data on village demographics, location and public good availability come from the Census of India 2001. *Public goods rank* is a village's percentile rank based on public good availability. We construct a public goods index by (i) constructing dummy variables indicating availability of public goods for five different categories, namely education, health, sanitation, communications and electricity, and (ii) taking the principal component of the 5 category indexes. Each village's public goods rank (0-1) is based on its public goods index score. All regressions include district fixed effects, and controls for the share of Scheduled Caste, Scheduled Tribe, General caste and tribal households, the total male and female population, and the fraction of adults. Standard errors are clustered at the constituency level. Dots represent the local average of the outcome variable, calculated within 1 percentage point bins of the running variable. Continuous lines are a 4th order polynomial fit.

**FIGURE 16.13:** George, 2019, descendant effects identified using close elections RD design (p. 41).

We find that the firm had a positive and persistent effect on living standards. Regions within the UFCo were 26 per cent less likely to be poor in 1973 than nearby counterfactual locations, with only 63 per cent of the gap closing over the following three decades. Company documents explain that a key concern at the time was to attract and maintain a sizable workforce, which induced the firm to invest heavily in local amenities that likely account for our result.

The model is:

$$y_{i,g,t} = \gamma_{\text{UFCo}_g} + f(\text{geographic location}_g) + X_{i,g,t}\beta + X_g\Gamma + \alpha_t + \epsilon_{i,g,t}.$$

In this model,  $y_{i,g,t}$  is the development outcome for a household  $i$  in census-block  $g$  and year  $t$ ;  $\gamma_{\text{UFCo}_g}$  is an indicator variable as to whether the census-block was in a UFCo area or not;  $f(\text{geographic location}_g)$  is a function of the latitude and longitude to adjust for geographic area;  $X_{i,g,t}$  is covariates for household  $i$ ;  $X_g$  is geographic characteristics for that census-block; and  $\alpha_t$  is a year fixed effect.

Again, there are a lot of different results but one is shown in Figure 16.14.

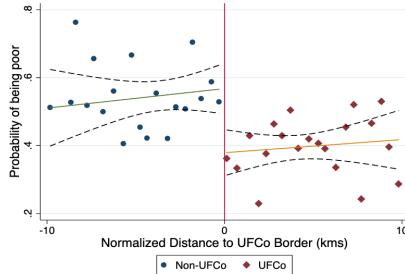
### 16.6.5 Implementation

Although they are fairly conceptually similar to work that we have done in the past, if you are wanting to use regression discontinuity in your work then you might like to consider a specialised package. The package `rdrobust` is a one recommendation, although there are others available and you should try those if you are interested. (The `rdd` package had been the go-to for a while, but seems to have been taken off CRAN recently. If you use RDD, then maybe just follow up to see if it comes back on as that one is pretty nice.)

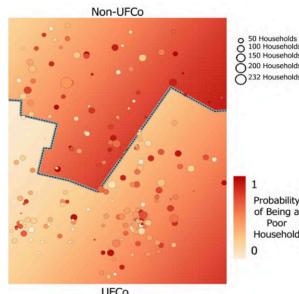
Let's look at our example using `rdrobust`.

```
library(rdrobust)
rdrobust(y = rdd_example_data$income,
 x = rdd_example_data$grade,
 c = 80, h = 2, all = TRUE) %>%
 summary()
#> Call: rdrobust
#>
#> Number of Obs. 1000
```

Figure 4: UFC EFFECT ON THE PROBABILITY OF BEING POOR



(a) Probability of being poor vs distance to the boundary



(b) Spatial distribution of households across space

*Notes:* Panel 3a shows a sharp discontinuity in the probability of being poor at the study boundary, with the probability being lower for households treated by the UFC (to the right). Slopes of the fitted lines are .011 (non-UFCo) and .008 (UFCo). Panel 3b shows the study boundary, with UFC territories being South. Each dot represents a census-block's centroid. Dot-color indicates the average outcome value for households, and dot-size represents the number of households in each census-block. As shown, lighter colors stand for better economic outcomes.

**FIGURE 16.14:** George, 2020, UFCo effect on the probability of being poor (p. 17).

#> BW type	Manual	
#> Kernel	Triangular	
#> VCE method	NN	
#>		
#> Number of Obs.	497	503
#> Eff. Number of Obs.	497	503
#> Order est. (p)	1	1
#> Order bias (q)	2	2
#> BW est. (h)	2.000	2.000
#> BW bias (b)	2.000	2.000
#> rho (h/b)	1.000	1.000

```
#> Unique Obs. 497 503
#>
#> =====
#> Method Coef. Std. Err. z P>|z| [95% C.I.]
#> =====
#> Conventional 1.974 0.143 13.783 0.000 [1.693 , 2.255]
#> Bias-Corrected 1.977 0.143 13.805 0.000 [1.696 , 2.258]
#> Robust 1.977 0.211 9.374 0.000 [1.564 , 2.390]
#> =====
```

### 16.6.6 Fuzzy RDD

The examples to this point have been ‘sharp’ RDD. That is, the threshold is strict. However, in reality, often the boundary is a little less strict. For instance, consider the drinking age. Although there is a legal drinking age, say 19. If we looked at the number of people who had drank, then it’s likely to increase in the few years leading up to that age. Perhaps you went to Australia where the drinking age is 18 and drank. Or perhaps you snuck into a bar when you were 17, etc.

In a sharp RDD setting, if you know the value of the forcing function then you know the outcome. For instance, if you get a grade of 80 then we know that you got an A-, but if you got a grade of 79 then we know that you got a B+. But with fuzzy RDD it is only known with some probability. We can say that a Canadian 19-year-old is more likely to have drunk alcohol than a Canadian 18 year old, but the number of Canadian 18-year-olds who have drunk alcohol is not zero.

It may be possible to deal with fuzzy RDD settings with appropriate choice of model or data. It may also be possible to deal with them using instrumental variables, which we cover in the next section.

### 16.6.7 Threats to validity

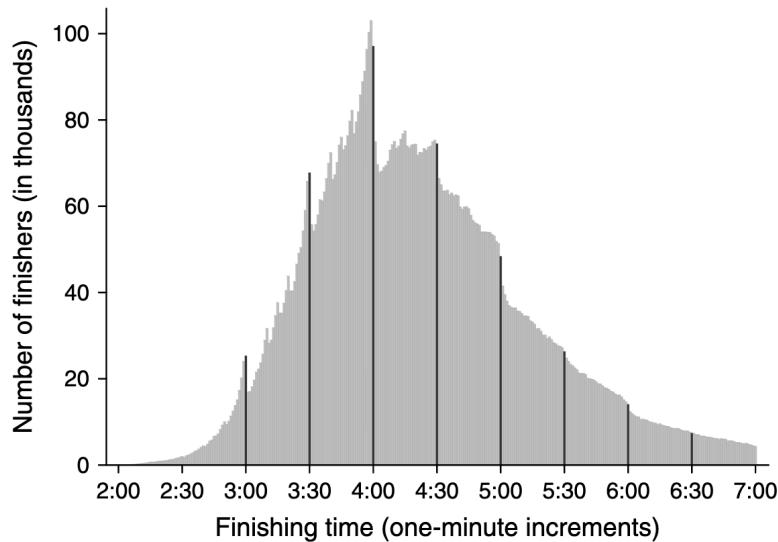
The continuity assumption is fairly important, but we cannot test this as it is based on a counterfactual. Instead we need to convince people of it. Ways to do this include:

- Using a test/train set-up.
- Trying different specifications (and be very careful if your results don’t broadly persist when just consider linear or quadratic functions).
- Considering different subsets of the data.
- Consider different windows.
- Be up-front about uncertainty intervals, especially in graphs.
- Discuss and assuage concerns about the possibility of omitted variables.

The threshold is also important. For instance, is there an actual shift or is there a non-linear relationship?

We want as ‘sharp’ an effect as possible, but if the thresholds are known, then they will be gamed. For instance, there is a lot of evidence that people run for certain marathon times, and we know that people aim for certain grades. Similarly, from the other side, it is a lot easier for an instructor to just give out As than it is to have to justify Bs. One way to look at this is to consider how ‘balanced’ the sample is on either side of the threshold. Do this by using histograms with appropriate bins, for instance Figure 16.15, which is from Allen et al. (2017).

**Figure 2.** Distribution of Marathon Finishing Times  
( $n = 9,789,093$ )



*Note.* The dark bars highlight the density in the 1-minute bin just before each 30-minute threshold.

**FIGURE 16.15:** Bunching around marathon times.

You need to really think about the possible effect of the decision around the choice of model. To see this consider the difference between a linear and polynomial.

```
some_data <-
 tibble(outcome = rnorm(n = 100, mean = 1, sd = 1),
 running_variable = c(1:100),
 location = "before")
```

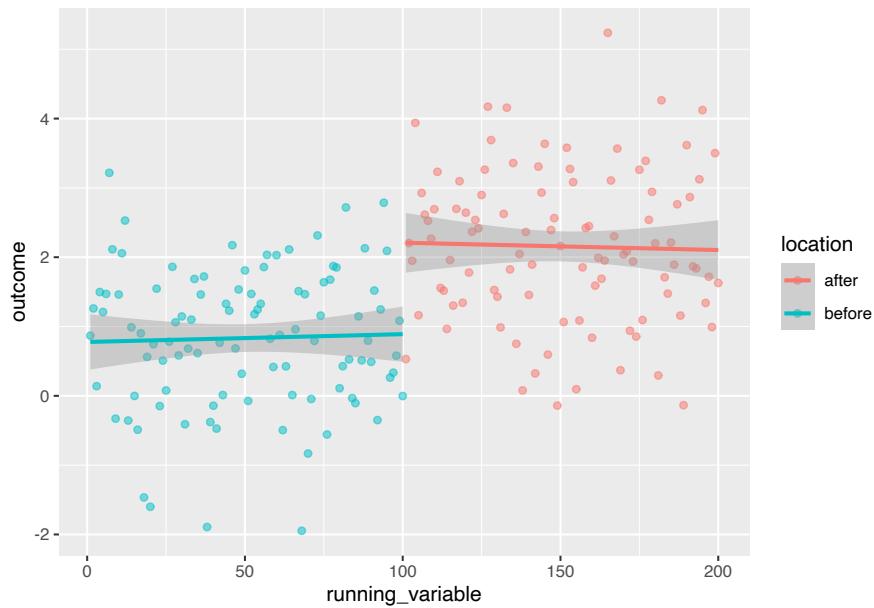
```

some_more_data <-
 tibble(outcome = rnorm(n = 100, mean = 2, sd = 1),
 running_variable = c(101:200),
 location = "after")

both <-
 rbind(some_data, some_more_data)

both %>%
 ggplot(aes(x = running_variable, y = outcome, color = location)) +
 geom_point(alpha = 0.5) +
 geom_smooth(formula = y~x, method = 'lm')

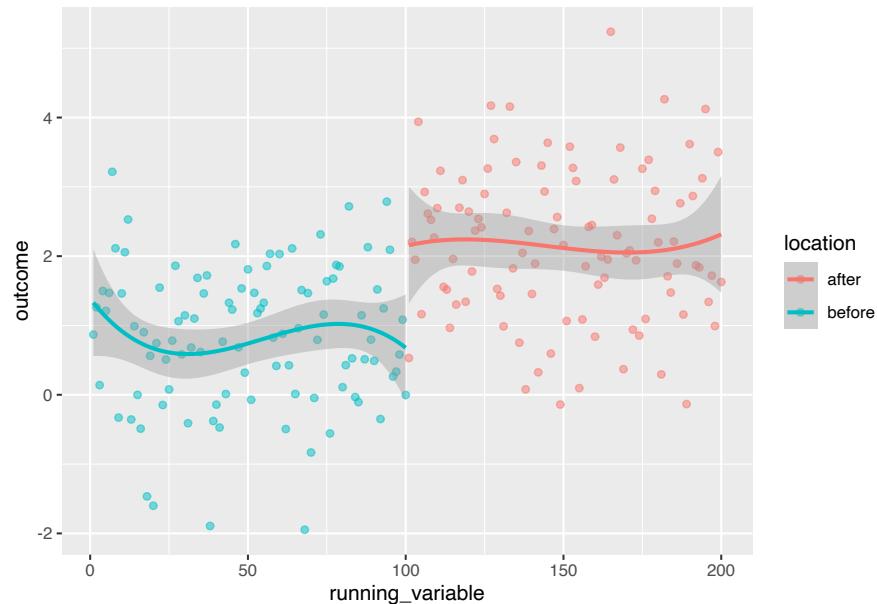
```



```

both %>%
 ggplot(aes(x = running_variable, y = outcome, color = location)) +
 geom_point(alpha = 0.5) +
 geom_smooth(formula = y ~ poly(x, 3), method = 'lm')

```



### 16.6.8 Weaknesses

- External validity may be hard - think about the A-/B+ example - do you think the findings generalise to B-/C+?
- The important responses are those that are close to the cut-off. So even if we have a whole bunch of B- and A+ students, they don't really help much. Hence we need a lot of data.
- There is a lot of freedom for the researcher, so open science best practice becomes vital.

## 16.7 Case study - Stiers, Hooghe, and Dassonneville, 2020

Paper: Stiers, D., Hooghe, M. and Dassonneville, R., 2020. Voting at 16: Does lowering the voting age lead to more political engagement? Evidence from a quasi-experiment in the city of Ghent (Belgium). *Political Science Research and Methods*, pp.1-8. Available at: <https://www.cambridge.org/core/journals/political-science-research-and-methods/article/voting-at-16-does-lowering-the-voting-age-lead-to-more-political-engagement-evidence-from-a-quasiexperiment-in>

the-city-of-ghent-belgium/172A2D9B75ECB66E98C9680787F302AD#fndtn-information

Data: <https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/J1FQW9>

---

## 16.8 Case study - Caughey, and Sekhon., 2011

Paper: Caughey, Devin, and Jasjeet S. Sekhon. "Elections and the regression discontinuity design: Lessons from close US house races, 1942–2008." Political Analysis 19.4 (2011): 385–408. Available at: <https://www.cambridge.org/core/journals/political-analysis/article/elections-and-the-regression-discontinuity-design-lessons-from-close-us-house-races-19422008/E5A69927D29BE682E012CAE9BFD8AEB7>

Data: <https://dataverse.harvard.edu/dataset.xhtml?persistentId=hdl:1902.1/16357&version=1.0>

---

## 16.9 Instrumental variables

### 16.9.1 Introduction

Instrumental variables (IV) is an approach that can be handy when we have some type of treatment and control going on, but we have a lot of correlation with other variables and we possibly don't have a variable that actually measures what we are interested in. So adjusting for observables will not be enough to create a good estimate. Instead we find some variable - the eponymous instrumental variable - that is:

1. correlated with the treatment variable, but
2. not correlated with the outcome.

This solves our problem because the only way the instrumental variable can have an effect is through the treatment variable, and so we are able to adjust our understanding of the effect of the treatment variable appropriately. The trade-off is that instrumental variables must satisfy a bunch of different assumptions, and that, frankly, they are difficult to identify *ex ante*. Nonetheless, when you are able to use them they are a powerful tool for speaking about causality.

The canonical instrumental variables example is smoking. These days we know

that smoking causes cancer. But because smoking is correlated with a lot of other variables, for instance, education, it could be that it was actually education that causes cancer. RCTs may be possible, but they are likely to be troublesome in terms of speed and ethics, and so instead we look for some other variable that is correlated with smoking, but not, in and of itself, with lung cancer. In this case, we look to tax rates, and other policy responses, on cigarettes. As the tax rates on cigarettes are correlated with the number of cigarettes that are smoked, but not correlated with lung cancer, other than through their impact on cigarette smoking, through them we can assess the effect of cigarettes smoked on lung cancer.

To implement instrumental variables we first regress tax rates on cigarette smoking to get some coefficient on the instrumental variable, and then (in a separate regression) regress tax rates on lung cancer to again get some coefficient on the instrumental variable. Our estimate is then the ratio of these coefficients. (Gelman and Hill, 2007, p. 219) describe this ratio as the ‘Wald estimate’.

Following the language of (Gelman and Hill, 2007, p. 216) when we use instrumental variables we make a variety of assumptions including:

- Ignorability of the instrument.
- Correlation between the instrumental variable and the treatment variable.
- Monotonicity.
- Exclusion restriction.

To summarise exactly what instrumental variables is about, I cannot do better than recommend the first few pages of the ‘Instrumental Variables’ chapter in Cunningham (2021), and this key paragraph in particular (by way of background, Cunningham has explained why it would have been impossible to randomly allocate ‘clean’ and ‘dirty’ water through a randomised controlled trial and then continues...):

---

Snow would need a way to trick the data such that the allocation of clean and dirty water to people was not associated with the other determinants of cholera mortality, such as hygiene and poverty. He just would need for someone or something to be making this treatment assignment for him.

Fortunately for Snow, and the rest of London, that someone or something existed. In the London of the 1800s, there were many different water companies serving different areas of the city. Some were served by more than one company. Several took their water from the Thames, which was heavily polluted by sewage. The service areas of such companies had much higher

rates of cholera. The Chelsea water company was an exception, but it had an exceptionally good filtration system. That's when Snow had a major insight. In 1849, Lambeth water company moved the intake point upstream along the Thames, above the main sewage discharge point, giving its customers purer water. Southwark and Vauxhall water company, on the other hand, left their intake point downstream from where the sewage discharged. Insofar as the kinds of people that each company serviced were approximately the same, then comparing the cholera rates between the two houses could be the experiment that Snow so desperately needed to test his hypothesis.

---

### 16.9.2 History

The history of instrumental variables is a rare statistical mystery, and [Stock and Trebbi \(2003\)](#) provide a brief overview. The method was first published in [Wright \(1928\)](#). This is a book about the effect of tariffs on animal and vegetable oil. So why might instrumental variables be important in a book about tariffs on animal and vegetable oil? The fundamental problem is that the effect of tariffs depends on both supply and demand. But we only know prices and quantities, so we don't know what is driving the effect. We can use instrumental variables to pin down causality.

Where it gets interesting, and becomes something of a mystery, is that the instrumental variables discussion is only in Appendix B. If you made a major statistical break-through would you hide it in an appendix? Further, Philip G. Wright, the book's author, had a son Sewall Wright, who had considerable expertise in statistics and the specific method used in Appendix B. Hence the mystery of Appendix B - did Philip or Sewall write it? Both [Cunningham \(2021\)](#) and [Stock and Trebbi \(2003\)](#) go into more detail, but on balance feel that it is likely that Philip did actually author the work.

### 16.9.3 Simulated example

Let's generate some data. We will explore a simulation related to the canonical example of health status, smoking, and tax rates. So we are looking to explain how healthy someone is based on the amount they smoke, via the tax rate on smoking. We are going to generate different tax rates by provinces. My understanding is that the tax rate on cigarettes is now pretty much the same in each of the provinces, but that this is fairly recent. So we'll pretend that Alberta had a low tax, and Nova Scotia had a high tax.

As a reminder, we are simulating data for illustrative purposes, so we need to

impose the answer that we want. When you actually use instrumental variables you will be reversing the process.

```
library(broom)
library(tidyverse)

set.seed(853)

number_of_observation <- 10000

iv_example_data <- tibble(person = c(1:number_of_observation),
 smoker = sample(x = c(0:1),
 size = number_of_observation,
 replace = TRUE))

```

Now we need to relate the number of cigarettes that someone smoked to their health. We'll model health status as a draw from the normal distribution, with either a high or low mean depending on whether the person smokes.

```
iv_example_data <-
 iv_example_data %>%
 mutate(health = if_else(smoker == 0,
 rnorm(n = n(), mean = 1, sd = 1),
 rnorm(n = n(), mean = 0, sd = 1)
)
)
So health will be one standard deviation higher for people who don't or barely smoke.
```

Now we need a relationship between cigarettes and the province (because in this illustration, the provinces have different tax rates).

```
iv_example_data <-
 iv_example_data %>%
 rowwise() %>%
 mutate(province = case_when(smoker == 0 ~ sample(x = c("Nova Scotia", "Alberta"),
 size = 1,
 replace = FALSE,
 prob = c(1/2, 1/2)),
 smoker == 1 ~ sample(x = c("Nova Scotia", "Alberta"),
 size = 1,
 replace = FALSE,
 prob = c(1/4, 3/4)))) %>%
ungroup()
```

```
iv_example_data <-
 iv_example_data %>%
 mutate(tax = case_when(province == "Alberta" ~ 0.3,
 province == "Nova Scotia" ~ 0.5,
 TRUE ~ 9999999
)
)

iv_example_data$tax %>% table()
#> .
#> 0.3 0.5
#> 6206 3794

head(iv_example_data)
```

```
person
smoker
health
province
tax
1
0
1.11
Alberta
0.3
2
1
-0.0831
Alberta
0.3
3
1
-0.0363
Alberta
```

0.3

4

0

2.48

Alberta

0.3

5

0

0.617

Alberta

0.3

6

0

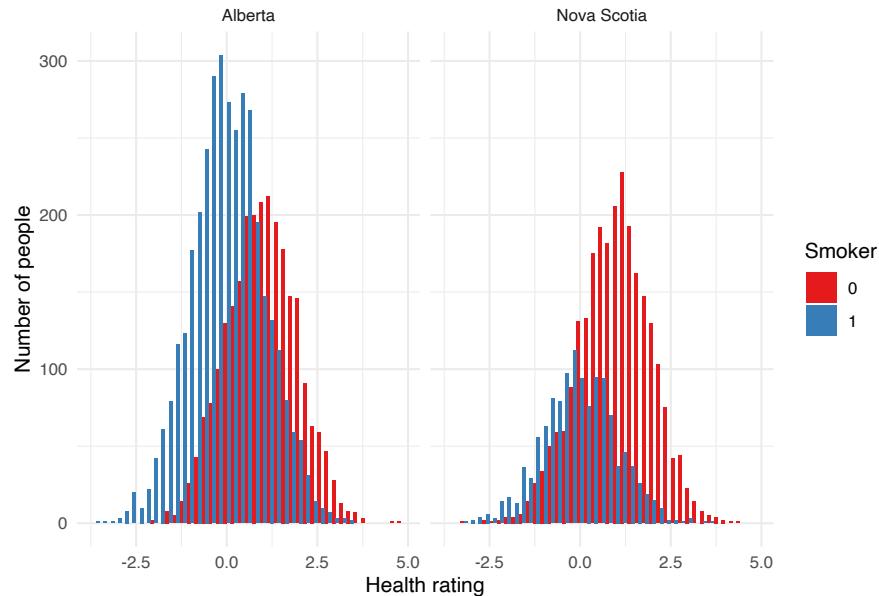
0.748

Nova Scotia

0.5

Now we can look at our data.

```
iv_example_data %>%
 mutate(smoker = as_factor(smoker)) %>%
 ggplot(aes(x = health, fill = smoker)) +
 geom_histogram(position = "dodge", binwidth = 0.2) +
 theme_minimal() +
 labs(x = "Health rating",
 y = "Number of people",
 fill = "Smoker") +
 scale_fill_brewer(palette = "Set1") +
 facet_wrap(vars(province))
```



Finally, we can use the tax rate as an instrumental variable to estimate the effect of smoking on health.

```
health_on_tax <- lm(health ~ tax, data = iv_example_data)
smoker_on_tax <- lm(smoker ~ tax, data = iv_example_data)

coef(health_on_tax)[["tax"]] / coef(smoker_on_tax)[["tax"]]
#> tax
#> -0.8554502
```

So we find, luckily, that if you smoke then your health is likely to be worse than if you don't smoke.

Equivalently, we can think of instrumental variables in a two-stage regression context.

```
first_stage <- lm(smoker ~ tax, data = iv_example_data)
health_hat <- first_stage$fitted.values
second_stage <- lm(health ~ health_hat, data = iv_example_data)

summary(second_stage)
#>
#> Call:
#> lm(formula = health ~ health_hat, data = iv_example_data)
```

```
#>
#> Residuals:
#> Min 1Q Median 3Q Max
#> -3.9867 -0.7600 0.0068 0.7709 4.3293
#>
#> Coefficients:
#> Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 0.91632 0.04479 20.46 <2e-16 ***
#> health_hat -0.85545 0.08911 -9.60 <2e-16 ***
#> ---
#> Signif. codes:
#> 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 1.112 on 9998 degrees of freedom
#> Multiple R-squared: 0.009134, Adjusted R-squared: 0.009034
#> F-statistic: 92.16 on 1 and 9998 DF, p-value: < 2.2e-16
```

#### 16.9.4 Implementation

As with regression discontinuity, although it is possible to use existing functions, it might be worth looking at specialised packages. Instrumental variables has a few moving pieces, so a specialised package can help keep everything organised, and additionally, standard errors need to be adjusted and specialised packages make this easier. The package `estimatr` is a recommendation, although there are others available and you should try those if you are interested. The `estimatr` package is from the same team as `DeclareDesign`.

Let's look at our example using `iv_robust()`.

```
library(estimatr)
iv_robust(health ~ smoker | tax, data = iv_example_data) %>%
 summary()
#>
#> Call:
#> iv_robust(formula = health ~ smoker | tax, data = iv_example_data)
#>
#> Standard error type: HC2
#>
#> Coefficients:
#> Estimate Std. Error t value Pr(>|t|) CI Lower
#> (Intercept) 0.9163 0.04057 22.59 3.163e-110 0.8368
#> smoker -0.8555 0.08047 -10.63 2.981e-26 -1.0132
#> CI Upper DF
```

```
#> (Intercept) 0.9958 9998
#> smoker -0.6977 9998
#>
#> Multiple R-squared: 0.1971 , Adjusted R-squared: 0.197
#> F-statistic: 113 on 1 and 9998 DF, p-value: < 2.2e-16
```

### 16.9.5 Assumptions

As discussed earlier, there are a variety of assumptions that are made when using instrumental variables. The two most important are:

1. Exclusion Restriction. This assumption is that the instrumental variable only affects the dependent variable through the independent variable of interest.
2. Relevance. There must actually be a relationship between the instrumental variable and the independent variable.

There is typically a trade-off between these two. There are plenty of variables that

When thinking about potential instrumental variables Cunningham (2021), p. 211, puts it brilliantly:

But, let's say you think you do have a good instrument. How might you defend it as such to someone else? A necessary but not a sufficient condition for having an instrument that can satisfy the exclusion restriction is if people are confused when you tell them about the instrument's relationship to the outcome. Let me explain. No one is going to be confused when you tell them that you think family size will reduce female labor supply. They don't need a Becker model to convince them that women who have more children probably work less than those with fewer children. It's common sense. But, what would they think if you told them that mothers whose first two children were the same gender worked less than those whose children had a balanced sex ratio? They would probably give you a confused look. What does the gender composition of your children have to do with whether a woman works?

It doesn't – it only matters, in fact, if people whose first two children are the same gender decide to have a third child. Which brings us back to the original point – people buy that family

size can cause women to work less, but they're confused when you say that women work less when their first two kids are the same gender. But if when you point out to them that the two children's gender induces people to have larger families than they would have otherwise, the person "gets it", then you might have an excellent instrument.

---

Relevance can be tested using regression and other tests for correlation. The exclusion restriction cannot be tested. You need to present evidence and convincing arguments. As Cunningham (2021) p. 225 says 'Instruments have a certain ridiculousness to them[.] That is, you know you have a good instrument if the instrument itself doesn't seem relevant for explaining the outcome of interest because that's what the exclusion restriction implies.'

#### 16.9.6 Conclusion

Instrumental variables is a useful approach because one can obtain causal estimates even without explicit randomisation. Finding instrumental variables used to be a bit of a white whale, especially in academia. However, I will leave the final (and hopefully motivating) word to Taddy (2019), p. 162:

---

As a final point on the importance of IV models and analysis, note that when you are on the inside of a firm—especially on the inside of a modern technology firm—explicitly randomised instruments are everywhere.... But it is often the case that decision-makers want to understand the effects of policies that are not themselves randomised but are rather downstream of the things being AB tested. For example, suppose an algorithm is used to predict the creditworthiness of potential borrowers and assign loans. Even if the process of loan assignment is never itself randomised, if the parameters in the machine learning algorithms used to score credit are AB tested, then those experiments can be used as instruments for the loan assignment treatment. Such 'upstream randomisation' is extremely common and IV analysis is your key tool for doing causal inference in that setting.'

---

## 16.10 Case study - Effect of Police on Crime

### 16.10.1 Overview

Here we'll use an example of [Levitt \(2002\)](#) that looks at the effect of police on crime. This is interesting because you might think, that more police is associated with lower crime. But, it could actually be the opposite, if more crime causes more police to be hired - how many police would a hypothetical country with no crime need? Hence there is a need to find some sort of instrumental variable that affects crime only through its relationship with the number of police (that is, not in and of itself, related to crime), and yet is also correlated with police numbers. [Levitt \(2002\)](#) suggests the number of firefighters in a city.

[Levitt \(2002\)](#) argues that firefighters are appropriate as an instrument, because '(f)actors such as the power of public sector unions, citizen tastes for government services, affirmative action initiatives, or a mayor's desire to provide spoils might all be expected to jointly influence the number of firefighters and police.' [Levitt \(2002\)](#) also argues that the relevance assumption is met by showing that 'changes in the number of police officers and firefighters within a city are highly correlated over time'.

In terms of satisfying the exclusion restriction, [Levitt \(2002\)](#) argues that the number of firefighters should not have a 'direct impact on crime.' However, it may be that there are common factors, and so [Levitt \(2002\)](#) adjusts for this in the regression.

### 16.10.2 Data

The dataset is based on 122 US cities between 1975 and 1995. Summary statistics are provided in Figure [16.16](#).

Source: [Levitt \(2002\)](#) p. 1,246.

### 16.10.3 Model

In the first stage [Levitt \(2002\)](#) looks at police as a function of firefighters, and a bunch of adjustment variables:

$$\ln(\text{Police}_{ct}) = \gamma \ln(\text{Fire}_{ct}) + X'_{ct}\Gamma + \lambda_t + \phi_c + \epsilon_{ct}.$$

The important part of this is the police and firefighters numbers which are on a per capita basis. There are a bunch of adjustment variables in  $X$  which includes things like state prisoners per capita, the unemployment rate, etc, as well as year dummy variables and fixed-effects for each city.

Having established the relationship between police and firefigths, [Levitt \(2002\)](#)

TABLE 1—SUMMARY STATISTICS

Variable	Mean	Standard deviation	
		Overall	Removing city fixed effects and year dummies
Police per capita	0.0033	0.0011	0.0003
Firefighters per capita	0.0016	0.0005	0.0001
Violent crime per capita	0.0136	0.0068	0.0022
Property crime per capita	0.0765	0.0191	0.0097
City population	2,005,602	2,539,372	105,809
Effective abortion rate (per 1,000 live births)	32.3	60.7	30.5
Unemployment rate	0.068	0.022	0.012
Real state income per capita (1999 dollars)	22,509	3,290	850
Percentage black	0.251	0.159	0.018
State prisoners per capita	0.0020	0.0011	0.0003

*Notes:* The unit of observation is a city-year pair. Data cover the period 1975–1995. To be included in the sample, a city had to have a population greater than 100,000 in 1973. Police, crime, and city population data are from the Federal Bureau of Investigation's *Uniform Crime Reports*. Firefighter data are from *Annual Survey of Governments*. The effective abortion rate is the weighted average of the abortion rate of crime-aged individuals, as calculated by Donohue and Levitt (2001). Unemployment data corresponds to a city's MSA and is from Department of Labor reports. State income per capita is from *Statistical Abstract of the United States*. The variable Percentage black is linearly interpolated between census years. The state prison population is from Bureau of Justice Statistics reports. There is a substantial amount of missing data for the firefighter variable. All values in the table are weighted by city population.

FIGURE 16.16: Summary statistics for Levitt 2002.

can then use the estimates of the number of police, based on the number of firefighters, to explain crime rates:

$$\Delta \ln(\text{Crime}_{ct}) = \beta_1 \ln(\text{Police}_{ct-1}) + X'_{ct} \Gamma + \Theta_c + \mu_{ct}.$$

The typical way to present instrumental variable results is to show both stages. Figure 16.17 shows the relationship between police and firefighters.

TABLE 2—THE RELATIONSHIP BETWEEN FIREFIGHTERS, POLICE, AND CRIME

Variable	First-stage estimates (dependent variable = $\ln(\text{Police per capita})$ )			Reduced-form estimates	
	(i)	(ii)	(iii)	Dependent variable: $\ln(\text{violent crime}$ per capita)	Dependent variable: $\ln(\text{property crime}$ per capita)
$\ln(\text{Firefighters per capita})$	0.251 (0.050)	0.236 (0.054)	0.206 (0.050)	-0.103 (0.050)	-0.118 (0.042)
$\ln(\text{Street and highway}$ workers per capita)	—	—	0.014 (0.014)	—	—
$\ln(\text{State prisoners per capita})$	—	-0.101 (0.022)	-0.077 (0.022)	-0.130 (0.036)	-0.255 (0.030)
Unemployment rate	—	0.571 (0.276)	0.265 (0.314)	-0.723 (0.365)	0.945 (0.277)
State income per capita ( $\times 10,000$ )	—	0.150 (0.004)	0.211 (0.005)	0.004 (0.006)	0.002 (0.005)
Effective abortion rate ( $\times 100$ )	—	0.033 (0.013)	0.045 (0.013)	-0.156 (0.025)	-0.127 (0.025)
$\ln(\text{City population})$	—	0.040 (0.040)	-0.014 (0.047)	0.161 (0.067)	-0.374 (0.068)
Percentage black	—	0.361 (0.204)	0.493 (0.264)	0.222 (0.334)	0.336 (0.271)
City-fixed effects and year dummies included?	yes	yes	yes	yes	yes
$R^2$ :	0.947	0.952	0.962	0.931	0.817
Number of observations:	2,032	2,032	1,445	2,005	2,032

*Notes:* The dependent variable is specified at the top of each column. All estimates are weighted least squares using city population as weights. Heteroscedasticity-robust standard errors are in parentheses. The unit of observation is a city-year pair for cities with population greater than 100,000 in 1973, for the period 1975–1995. Sample size varies due to missing data, especially for municipal streets and highway workers. In the final two columns, the firefighter and prison population variables are once-lagged. For further description of the variables used and data sources, see the notes to Table 1.

**FIGURE 16.17:** The relationship between firefighters, police and crime.

Source: Levitt (2002) p. 1,247.

And then Figure 16.18 shows the relationship between police and crime, where it is the IV results that are the ones of interest.

Source: Levitt (2002) p. 1,248.

#### 16.10.4 Discussion

The key finding of Levitt (2002) is that there is a negative effect of the number of police on the amount of crime.

There are a variety of points that I want to raise in regard to this paper. They will come across as a little negative, but this is mostly just because this a paper from 2002, that I am reading today, and so the standards have changed.

1. It's fairly remarkable how reliant on various model specifications the results are. The results bounce around a fair bit and that's just

TABLE 3—THE IMPACT OF POLICE ON CRIME

Variable	Violent crime			Property crime		
	OLS	OLS	IV	OLS	OLS	IV
In(Police per capita) <sub>t-1</sub>	0.562 (0.056)	-0.076 (0.061)	-0.435 (0.231)	0.113 (0.038)	-0.218 (0.052)	-0.501 (0.235)
In(State prisoners per capita) <sub>t-1</sub>	0.250 (0.039)	-0.131 (0.036)	-0.171 (0.044)	0.189 (0.030)	-0.273 (0.028)	-0.305 (0.037)
Unemployment rate	3.573 (0.473)	-0.741 (0.365)	-0.480 (0.404)	1.283 (0.312)	1.023 (0.274)	1.231 (0.326)
State income per capita ( $\times 10,000$ )	0.050 (0.005)	-0.003 (0.006)	0.003 (0.007)	0.010 (0.003)	0.005 (0.004)	0.009 (0.006)
Effective abortion rate ( $\times 100$ )	-0.214 (0.045)	-0.150 (0.023)	-0.141 (0.025)	-0.184 (0.020)	-0.118 (0.021)	-0.111 (0.024)
In(City population)	0.072 (0.012)	0.203 (0.063)	0.178 (0.067)	-0.064 (0.006)	-0.333 (0.063)	-0.355 (0.066)
Percentage black	0.627 (0.074)	0.233 (0.334)	0.398 (0.345)	-0.136 (0.057)	0.411 (0.271)	0.517 (0.291)
City-fixed effects and year dummies included?	only year dummies	yes	yes	only year dummies	yes	yes
R <sup>2</sup> :	0.601	0.930	—	0.238	0.819	—
Number of observations:	2,005	2,005	2,005	2,032	2,032	2,032

*Notes:* The dependent variable is listed at the top of each column. The police and prison variables are once-lagged. Heteroscedasticity-robust standard errors are in parentheses. Columns (1), (2), (4), and (5) are estimated using weighted least squares, with city populations as weights. Columns (3) and (6) are instrumental variables estimates using the once-lagged, logged number of firefighters as an instrument for the once-lagged number of police. With the exception of columns (1) and (4), other regressions include both city-fixed effects and year dummies. See the notes to Table 1 for further description of the variables and the data sources.

**FIGURE 16.18:** The impact of police on crime.

the ones that are reported. Chances are there are a bunch of other results that were not reported, but it would be of interest to see their impact.

2. On that note, there is fairly limited model validation. This is probably something that I am more aware of these days, but it seems likely that there is a fair degree of over-fitting here.
3. Levitt (2002) is actually a response, after another researcher, McCrary (2002), found some issues with the original paper: Levitt (1997). While Levitt appears quite decent about it, it is jarring to see that Levitt was thanked by McCrary (2002) for providing ‘both the data and computer code.’ What if Levitt had not been decent about providing the data and code? Or what if the code was unintelligible? In some ways it is nice to see how far that we have come - the author of a similar paper these days would be forced to make their code and data available as part of the paper, we wouldn’t have to ask them for it. But it reinforces the importance of open data and reproducible science.

---

**16.11 Exercises and tutorial****16.11.1 Exercises****16.11.2 Tutorial**

# 17

---

## Multilevel regression with post-stratification

---

**STATUS:** Under construction.

### Required material

- Read *Analyzing name changes after marriage using a non-representative survey*, (Alexander, 2019a).
- Read Chapter 17 of *Regression and Other Stories*, (Gelman et al., 2020).
- Read *An introduction to multilevel regression and post-stratification for estimating constituency opinion*, (Hanretty, 2020).
- Read the Introduction from *Estimating State Public Opinion With Multi-Level Regression and Poststratification using R*, (Kastellec et al., 2016).
- Read *Using sex and gender in survey adjustment*, (Kennedy et al., 2020).
- Read *MRP with rstanarm*, (Kennedy and Gabry, 2020).
- Read *Know your population and know your model: Using model-based regression and poststratification to generalize findings beyond the observed sample*, (Kennedy and Gelman, 2020).
- Read *Forecasting elections with non-representative polls*, (Wang et al., 2015).
- Read Chapter 17 of *Sampling Theory and Practice*, (Wu and Thompson, 2020).
- Watch *Statistical Models of Election Outcomes*, (Gelman, 2020).
- Listen to *Episode 248: Are Democrats being irrational? (David Shor)*, (Galef, 2020).

### Recommended reading

- Arnold, Jeffrey B., 2018, ‘Simon Jackman’s Bayesian Model Examples in Stan’, Ch 13, 7 May, <https://jrnold.github.io/bugs-examples-in-stan/campaign.html>.
- Cohn, Nate, 2016, ‘We Gave Four Good Pollsters the Same Raw Data. They Had Four Different Results’, *The New York Times*, The Upshot, 20 September, <https://www.nytimes.com/interactive/2016/09/20/upshot/the-error-the-polling-world-rarely-talks-about.html>.
- Edelman, M., Vittert, L., & Meng, X.-L., 2021, ‘An Interview with Murray Edelman on the History of the Exit Poll’, *Harvard Data Science Review*, <https://doi.org/10.1162/99608f92.3a25cd24> <https://hdsr.mitpress.mit.edu/pub/fekmqbv4/release/2>.

- Gelman, Andrew, and Julia Azari, 2017, ‘19 things we learned from the 2016 election’, *Statistics and Public Policy*, 4 (1), pp. 1-10.
- Gelman, Andrew, Jessica Hullman, and Christopher Wlezien, 2020, ‘Information, incentives, and goals in election forecasts’, 8 September, available at: [http://www.stat.columbia.edu/~gelman/research/unpublished/forecast\\_incentives3.pdf](http://www.stat.columbia.edu/~gelman/research/unpublished/forecast_incentives3.pdf)
- Gelman, Andrew, Merlin Heidemanns, and Elliott Morris, 2020, ‘2020 US POTUS model’, The Economist, freely available: <https://github.com/TheEconomist/us-potus-model>.
- Ghitza, Yair, and Andrew Gelman, 2013, ‘Deep Interactions with MRP: Election Turnout and Voting Patterns Among Small Electoral Subgroups’, *American Journal of Political Science*, 57 (3), pp. 762-776.
- Ghitza, Yair, and Andrew Gelman, 2020, ‘Voter Registration Databases and MRP: Toward the Use of Large-Scale Databases in Public Opinion Research’, *Political Analysis*, pp. 1-25.
- Imai, Kosuke, 2017, Quantitative Social Science: An Introduction, Princeton University Press, Ch 4.1, and 5.3.
- Jackman, Simon, 2005, ‘Pooling the polls over an election campaign’, *Australian Journal of Political Science*, 40 (4), pp. 499-517.
- Jackman, Simon, Shaun Ratcliff and Luke Mansillo, 2019, ‘Small area estimates of public opinion: Model-assisted post-stratification of data from voter advice applications’, 4 January, <https://www.cambridge.org/core/membership/services/aop-file-manager/file/5c2f6ebb7cf9ee1118d11c0a/APMM-2019-Simon-Jackman.pdf>.
- Lauderdale, Ben, Delia Bailey, Jack Blumenau, and Doug Rivers, 2020, ‘Model-based pre-election polling for national and sub-national outcomes in the US and UK’, *International Journal of Forecasting*, 36 (2), pp. 399-413.
- Leigh, Andrew, and Justin Wolfers, 2006, ‘Competing approaches to forecasting elections: Economic models, opinion polling and prediction markets’, *Economic Record*, 82 (258), pp.325-340.
- Nickerson, David W., and Todd Rogers, 2014, ‘Political campaigns and big data’, *Journal of Economic Perspectives*, 28 (2), pp. 51-74.
- Shirani-Mehr, Houshmand, David Rothschild, Sharad Goel, and Andrew Gelman, 2018, ‘Disentangling bias and variance in election polls’, *Journal of the American Statistical Association*, 113 (522), pp. 607-614.

### Recommended viewing

- Jackman, Simon, 2020, ‘The triumph of the quants?: Model-based poll aggregation for election forecasting’, *Ihaka Lecture Series*, <https://youtu.be/MvGYsKIsLFs>.

### Key libraries

- `brms`
- `broom`
- `gtsummary`

- `haven`
- `labelled`
- `lme4`
- `modelsummary`
- `rstanarm`
- `tidybayes`
- `tidyverse`

### Quiz

1. Your Mum asked you what you've been learning this term. You decide to tell her about multilevel regression with post-stratification (MRP). Please explain what MRP is. Your Mum has a university-education, but has not necessarily taken any statistics, so you will need to explain any technical terms that you use. [Please write two or three paragraphs; strong answers would be clear about both strengths and weaknesses.]
2. With respect to Wang et al. (2015): Why is this paper interesting? What do you like about this paper? What do you wish it did better? To what extent can you reproduce this paper? [Please write one or two paragraphs about each aspect.]
3. With respect to Wang et al. (2015), what is not a feature they mention election forecasts need?
  - a. Explainable.
  - b. Accurate.
  - c. Cost-effective.
  - d. Relevant.
  - e. Timely.
4. With respect to Wang et al. (2015), what is a weakness of MRP?
  - a. Detailed data requirement.
  - b. Allows use of biased data.
  - c. Expensive to conduct.
5. With respect to Wang et al. (2015), what is concerning about the Xbox sample?
  - a. Non-representative.
  - b. Small sample size.
  - c. Multiple responses from the same respondent.
6. I am interested in studying how voting intentions in the 2020 US presidential election vary by an individual's income. I set up a logistic regression model to study this relationship. In my study, some possible independent variables would be: [Please check all that apply.]
  - a. Whether the respondent is registered to vote (yes/no).
  - b. Whether the respondent is going to vote for Biden (yes/no).
  - c. The race of the respondent (white/not white).

- d. The respondent's marital status (married/not).
7. Please think about Cohn (2016) Why is this type of exercise not carried out more? Why do you think that different groups, even with the same background and level of quantitative sophistication, could have such different estimates even when they use the same data? [Please write a paragraph or two about each aspect.]
8. When we think about multilevel regression with post-stratification, what are the key assumptions that we are making? [Please write one or two paragraphs about each aspect.]
9. I train a model based on a survey, and then post-stratify it using the 2020 ACS dataset. What are some of the practical considerations that I may have to contend when I am doing this? [Please write a paragraph each about at least three considerations.]
10. In a similar manner to Ghitza and Gelman (2020) pretend you've got access to a US voter file record from a private company. You train a model on the 2020 US CCES, and post-stratify it, on an individual-basis, based on that voter file.
- a. Could you please put-together a datasheet for the voter file dataset following Gebru et al. (2020)? As a reminder, datasheets accompany datasets and document 'motivation, composition, collection process, recommended uses,' among other aspects.
  - b. Could you also please put together a model card for your model, following Mitchell et al. (2019)? As a reminder, model cards are deliberately straight-forward one- or two-page documents that report aspects such as: model details; intended use; metrics; training data; ethical considerations; as well as caveats and recommendations (Mitchell et al., 2019).
  - c. Could you please discuss three ethical aspects around the features that you are using in your model? [Please write a paragraph or two for each point.]
  - d. Could you please detail the protections that you would put in place in terms of the dataset, the model, and the predictions?
11. If I have a model output from `lm()` called 'my\_model\_output' how can I use `modelsummary` to display the output (assume the package has been loaded) [please select all that apply]?
- a. `modelsummary::modelsummary(my_model_output)`
  - b. `modelsummary(my_model_output)`
  - c. `my_model_output %>% modelsummary()`
  - d. `my_model_output %>% modelsummary(statistic = NULL)`
12. Which of the following are examples of linear models [please select all that apply]?
- a. `lm(y ~ x_1 + x_2 + x_3, data = my_data)`
  - b. `lm(y ~ x_1 + x_2^2 + x_3, data = my_data)`
  - c. `lm(y ~ x_1 * x_2 + x_3, data = my_data)`

- d. `lm(y ~ x_1 + x_1^2 + x_2 + x_3, data = my_data)`
13. Consider a situation in which you have a survey dataset with these age-groups: 18-29; 30-44; 45- 60; and 60+. And a post-stratification dataset with these age-groups: 18-24; 25-29; 30-34; 35-39; 40-44; 45-49; 50-54; 55-59; and 60+. What approach would you take to bringing these together? [Please write a paragraph.]
14. Consider a situation in which you again have a survey dataset with these age-groups: 18-29; 30-44; 45- 60; and 60+. But this time the post-stratification dataset has these age-groups: 18-34; 35-49; 50-64; and 65+. What approach would you take to bringing these together? [Please write a paragraph.]
15. Please consider [Kennedy et al. \(2020\)](#). What are some statistical facets when considering a survey focused on gender, with a post-stratification survey that is not? [Please check all that apply.]
- Impute all non-male as female
  - Estimate gender using auxiliary information
  - Impute population
  - Impute sample values
  - Model population distribution using auxiliary data
  - Remove all non-binary respondents
  - Remove respondents
  - Assume population distribution
16. Please consider [Kennedy et al. \(2020\)](#). What are some ethical facets when considering a survey focused on gender, with a post-stratification survey that is not? [Please check all that apply.]
- Impute all non-male as female
  - Estimate gender using auxiliary information
  - Impute population
  - Impute sample values
  - Model population distribution using auxiliary data
  - Remove all non-binary respondents
  - Remove respondents
  - Assume population distribution
17. Please consider [Kennedy et al. \(2020\)](#). How do they define ethics?
- Respecting the perspectives and dignity of individual survey respondents.
  - Generating estimates of the general population and for subpopulations of interest.
  - Using more complicated procedures only when they serve some useful function.

## 17.1 Introduction

---

[The Presidential election of] 2016 was the largest analytics failure in US political history.

David Shor, 13 August 2020

---

Multilevel regression with post-stratification (MRP) is a popular way to adjust non-representative samples to better analyse opinion and other survey responses.<sup>1</sup> It uses a regression model to relate individual-level survey responses to various characteristics and then rebuilds the sample to better match the population. In this way MRP can not only allow a better understanding of responses, but also allow us to analyse data that may otherwise be unusable. However, it can be a challenge to get started with MRP as the terminology may be unfamiliar, and the data requirements can be onerous.

Let's say that we have a biased survey. Maybe we conducted a survey about computers at an academic seminar, so folks with post-graduate degrees are likely over-represented. We are nonetheless interested in making claims about the population. Let's say that we found 37.5 per cent of our respondents prefer Macs. One way forward is to just ignore the bias and say that '37.5 per cent of people prefer Macs'. Another way is to say, well 50 per cent of our respondents with a post-graduate degree prefer Macs, and of those without a post-graduate degree, 25 per cent prefer Macs. If we knew what proportion of the broader population has post-graduate degree, let's assume 10 per cent, then we could conduct re-weighting, or post-stratification, as follows:  $0.5 * 0.1 + 0.25 * 0.9 = 0.275$ , and so our estimate is that 27.5 per cent of people prefer Macs. MRP is a third approach, and uses a model to help do that re-weighting. So we use logistic regression to estimate the relationship between preferring Macs and highest educational attainment in our survey. We then apply that relationship to population dataset.

MRP is a handy approach when dealing with survey data. Hanretty (2020) puts it well when he says 'MRP is used because the alternatives are either very poor or very expensive.. Essentially, it trains a model based on the survey,

---

<sup>1</sup>I'd like to acknowledge and thank Lauren Kennedy and Monica Alexander<sup>2</sup>, through whose generous sharing of code, data, and countless conversations my thoughts about MRP have developed.

and then applies that trained model to another dataset. There are two main, related, advantages:

- 1) It can allow us to ‘re-weight’ in a way that includes uncertainty front-of-mind and isn’t as hamstrung by small samples. The alternative way to deal with having a small sample is to either go and gather more data or throw it away.
- 2) It can allow us to use broad surveys to speak to subsets. As Hanretty (2020) says ‘A poor alternative [to using MRP] is simply splitting a large sample into (much) smaller geographic subsamples. This is a poor alternative because there is no guarantee that a sample which is representative at the national level will be representative when it is broken down into smaller groups.’

From a practical perspective, it tends to be less expensive to collect non-probability samples and so there are benefits of being able to use these types of data. That said, it is not a magic-bullet and the laws of statistics still apply. We will have larger uncertainty around our estimates and they will still be subject to all the usual biases. As Lauren Kennedy<sup>3</sup> points out, ‘MRP has traditionally been used in probability surveys and had potential for non-probability surveys, but we’re not sure of the limitations at the moment.’ It’s an exciting area of research in both academia and industry.

The workflow that you need for MRP is straight-forward, but the details and tiny decisions that have to be made at each step can become overwhelming. The point that you need to keep in mind is that you are trying to create a relationship between two datasets using a statistical model, and so you need to establish similarity between the two datasets in terms of their variables and levels. The steps are:

- 1) gather and prepare the survey dataset, thinking about what is needed for coherence with the post-stratification dataset;
- 2) gather and prepare the post-stratification dataset thinking about what is needed for coherence with the survey dataset;
- 3) model the variable of interest from the survey using independent variables and levels that are available in both the survey and the post-stratification datasets;
- 4) apply the model to the post-stratification data.

In these notes, we begin with simulating a situation in which we pretend that we know the features of the population. We then move to a famous example of MRP that used survey data from the Xbox platform and exit poll data to forecast the 2012 US election. We will then move to examples from the

---

<sup>3</sup><https://jazzystats.com>

Australian political situation. We will then discuss some features to be aware of when conducting MRP.

## 17.2 Simulation - Toddler bedtimes

### 17.2.1 Construct a population

To get started we will simulate some data from a population that has various properties, take a biased sample, and then conduct MRP to demonstrate how we can get those properties back. We are going to have two ‘explanatory variables’ - age-group and toilet-trained - and one dependent variable - bedtime. Bed-time will increase as age-group increases, and will be later for children that are toilet-trained, compared with those that are not. To be clear, in this example we will ‘know’ the ‘true’ features of the population, but this isn’t something that occurs when we use real data - it is just to help you understand what MRP is doing. We’re going to rely heavily on the `tidyverse` package (Wickham et al., 2019a).

```
Uncomment this (by deleting the #) if you need to install the packages
install.packages('tidyverse')
library(tidyverse)

This helps reproducibility
It makes it more likely that you're able to get the same random numbers as in this example.
set.seed(853)

One million people in our population.
size_of_population <- 1000000

population_for_mrp_example <-
 tibble(age_group = sample(x = c(1:3), # Draw from any of 1, 2, 3.
 size = size_of_population,
 replace = TRUE # After you draw a number, allow that number to be drawn again),
 toilet_trained = sample(x = c(0, 1),
 size = size_of_population,
 replace = TRUE),
 noise = rnorm(size_of_population, mean = 0, sd = 1),
 bed_time = 5 + 0.5 * age_group + 1 * toilet_trained + noise, # Make bedtime a linear function of age_group and toilet_trained
) %>%
 select(-noise) %>%
```

```

 mutate(age_group = as_factor(age_group),
 toilet_trained = as_factor(toilet_trained)
)

population_for_mrp_example %>%
 head()
#> # A tibble: 6 x 3
#> age_group toilet_trained bed_time
#> <fct> <fct> <dbl>
#> 1 1 0 5.74
#> 2 2 1 6.48
#> 3 1 0 6.53
#> 4 1 1 5.39
#> 5 1 1 8.40
#> 6 3 0 6.54

```

At this point, Figure 17.1 provides invaluable advice (thank you to A Mahfouz).

That is, as always, when we have a dataset, we first try to plot it to better understand what is going on (as there are a million points, I'll just grab the first 1,000 so that it plots nicely).

```

population_for_mrp_example %>%
 slice(1:1000) %>%
 ggplot(aes(x = age_group, y = bed_time)) +
 geom_jitter(aes(color = toilet_trained),
 alpha = 0.4,
 width = 0.1,
 height = 0) +
 labs(x = "Age-group",
 y = "Bed time",
 color = "Toilet trained") +
 theme_classic() +
 scale_color_brewer(palette = "Set1")

```

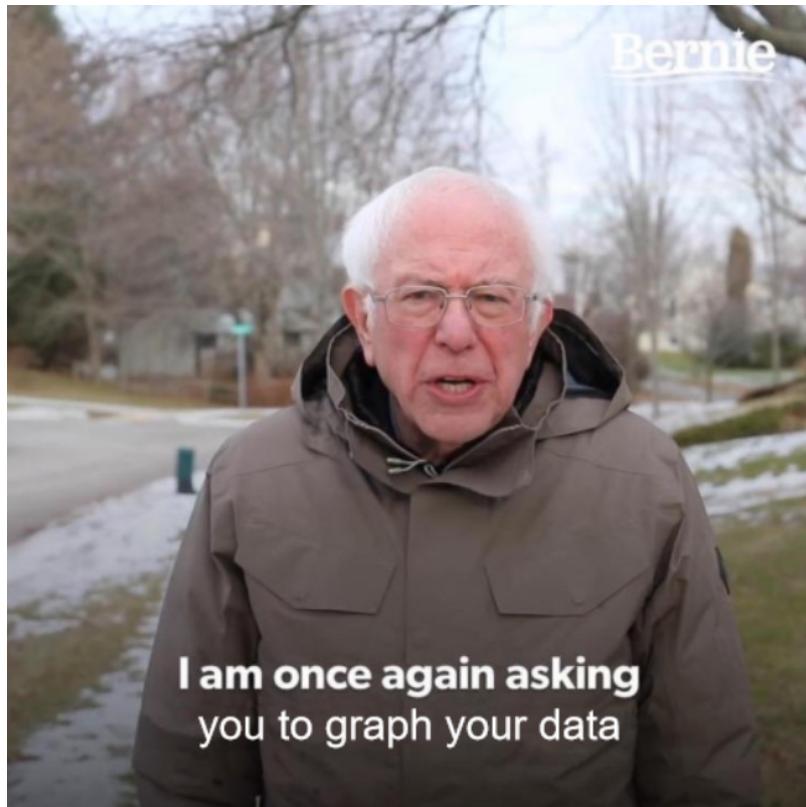
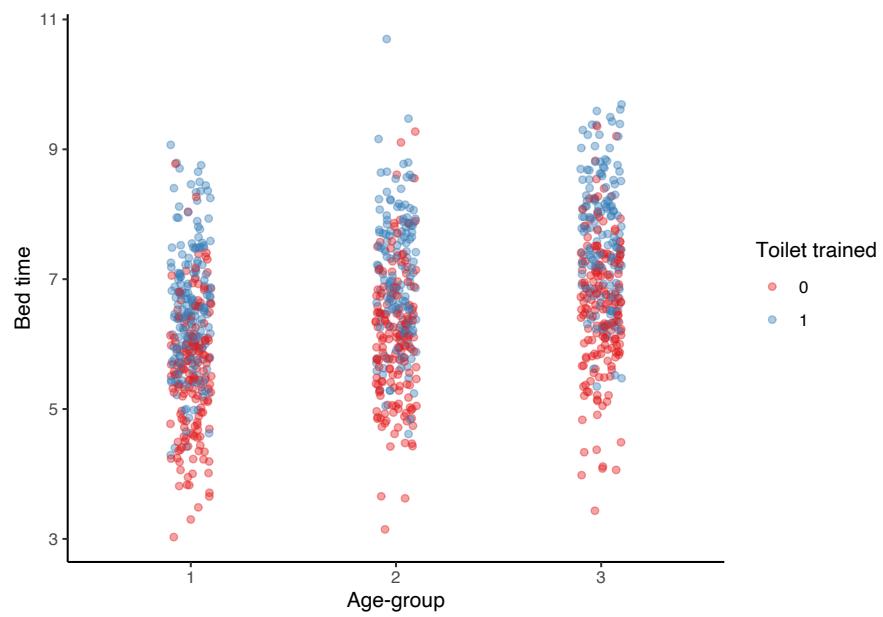


FIGURE 17.1: What does Bernie ask us to do?



And we can also work out what the ‘truth’ is for the information that we are interested in (remembering that we’d never actually know this when we move away from simulated examples).

```
population_for_mrp_example_summarised <-
 population_for_mrp_example %>%
 group_by(age_group, toilet_trained) %>%
 summarise(median_bed_time = median(bed_time))

population_for_mrp_example_summarised %>%
 knitr::kable(digits = 2,
 col.names = c("Age-group", "Is toilet trained", "Average bed time"))
```

Age-group	Is toilet trained	Average bed time
1	0	5.50
1	1	6.50
2	0	6.00
2	1	7.00
3	0	6.50
3	1	7.51

### 17.2.2 Get a biased sample from it

Now we want to pretend that we have some survey that has a biased sample. We’ll allow that it over-samples children that are younger and those that are not toilet-trained. For instance, perhaps we gathered our sample based on the records of a paediatrician, so it’s more likely that they will see this biased sample of children. We are interested in knowing what proportion of children are toilet-trained at various age-groups.

```
Thanks to Monica Alexander
set.seed(853)

Add a weight for each 'type' (has to sum to one)
population_for_mrp_example <-
 population_for_mrp_example %>%
 mutate(weight =
 case_when(toilet_trained == 0 & age_group == 1 ~ 0.7,
 toilet_trained == 0 ~ 0.1,
 age_group %in% c(1, 2, 3) ~ 0.2
),
 id = 1:n()
)
```

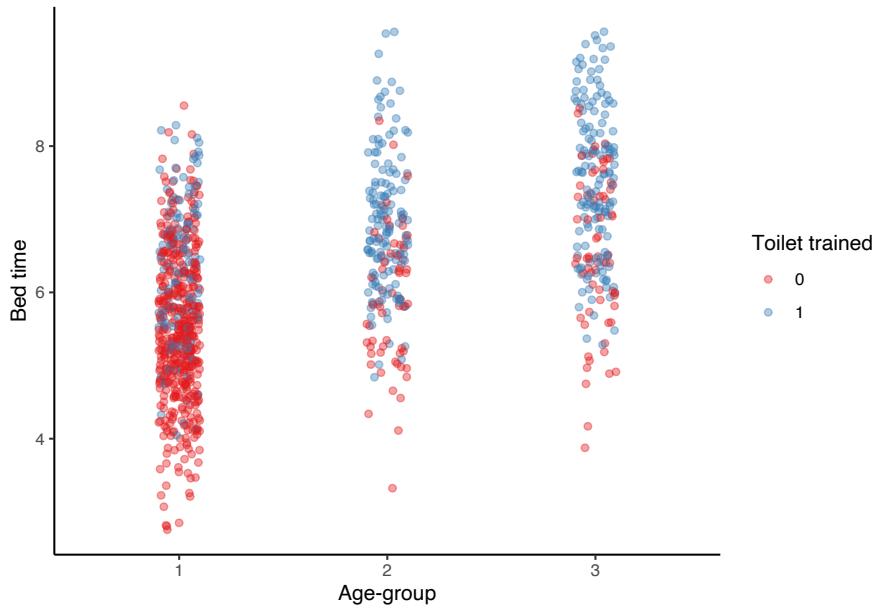
```
get_these <-
 sample(
 x = population_for_mrp_example$id,
 size = 1000,
 prob = population_for_mrp_example$weight
)

sample_for_mrp_example <-
 population_for_mrp_example %>%
 filter(id %in% get_these) %>%
 select(-weight, -id)

Clean up
poststratification_dataset <-
 population_for_mrp_example %>%
 select(-weight, -id)
```

And we can plot those also.

```
sample_for_mrp_example %>%
 mutate(toilet_trained = as_factor(toilet_trained)) %>%
 ggplot(aes(x = age_group, y = bed_time)) +
 geom_jitter(aes(color = toilet_trained), alpha = 0.4, width = 0.1, height = 0) +
 labs(x = "Age-group",
 y = "Bed time",
 color = "Toilet trained") +
 theme_classic() +
 scale_color_brewer(palette = "Set1")
```



It's pretty clear that our sample has a different bedtime than the overall population, but let's just do the same exercise as before to look at the median, by age and toilet-trained status.

```
sample_for_mrp_example_summarized <-
 sample_for_mrp_example %>%
 group_by(age_group, toilet_trained) %>%
 summarise(median_bed_time = median(bed_time))

sample_for_mrp_example_summarized %>%
 knitr::kable(digits = 2,
 col.names = c("Age-group", "Is toilet trained", "Average bed time"))
```

Age-group	Is toilet trained	Average bed time
1	0	5.41
1	1	6.35
2	0	5.89
2	1	6.85
3	0	6.49
3	1	7.62

	Model 1
(Intercept)	5.47 (0.04)
age_group2	0.54 (0.09)
age_group3	1.15 (0.09)
toilet_trained1	0.91 (0.07)
Num.Obs.	1000
R2	0.373
R2 Adj.	0.371
AIC	2839.3
BIC	2863.8
Log.Lik.	−1414.630
F	197.594

### 17.2.3 Model the sample

We will quickly train a model based on the (biased) survey. We'll use `modelsummary` (Arel-Bundock, 2021a) to format our estimates.

```
library(modelsummary)

mrp_example_model <-
 sample_for_mrp_example %>%
 lm(bed_time ~ age_group + toilet_trained, data = .)

mrp_example_model %>%
 modelsummary::modelsummary(fmt = 2)
```

This is the ‘multilevel regression’ part of the MRP (although this isn’t really a multilevel model just to keep things simple for now).

### 17.2.4 Get a post-stratification dataset

Now we will use a post-stratification dataset to get some estimates of the number in each cell. We typically use a larger dataset that may more closely reflect the population. In the US a popular choice is the ACS, while in other countries we typically have to use the census.

In this simulation example, I’ll just take a 10 per cent sample from the population and use that as our post-stratification dataset.

```

set.seed(853)

poststratification_dataset <-
 population_for_mrp_example %>%
 slice(1:100000) %>%
 select(-bed_time)

poststratification_dataset %>%
 head()
#> # A tibble: 6 x 4
#> age_group toilet_trained weight id
#> <fct> <fct> <dbl> <int>
#> 1 1 0 0.7 1
#> 2 2 1 0.2 2
#> 3 1 0 0.7 3
#> 4 1 1 0.2 4
#> 5 1 1 0.2 5
#> 6 3 0 0.1 6

```

In an ideal world we have individual-level data in our post-stratification dataset (that's the case above). In that world we can apply our model to each individual. The more likely situation, in reality, is that we just have counts by groups, so we're going to try to construct an estimate for each group.

```

poststratification_dataset_grouped <-
 poststratification_dataset %>%
 group_by(age_group, toilet_trained) %>%
 count()

poststratification_dataset_grouped %>%
 head()
#> # A tibble: 6 x 3
#> # Groups: age_group, toilet_trained [6]
#> age_group toilet_trained n
#> <fct> <fct> <int>
#> 1 1 0 16766
#> 2 1 1 16649
#> 3 2 0 16801
#> 4 2 1 16617
#> 5 3 0 16625
#> 6 3 1 16542

```

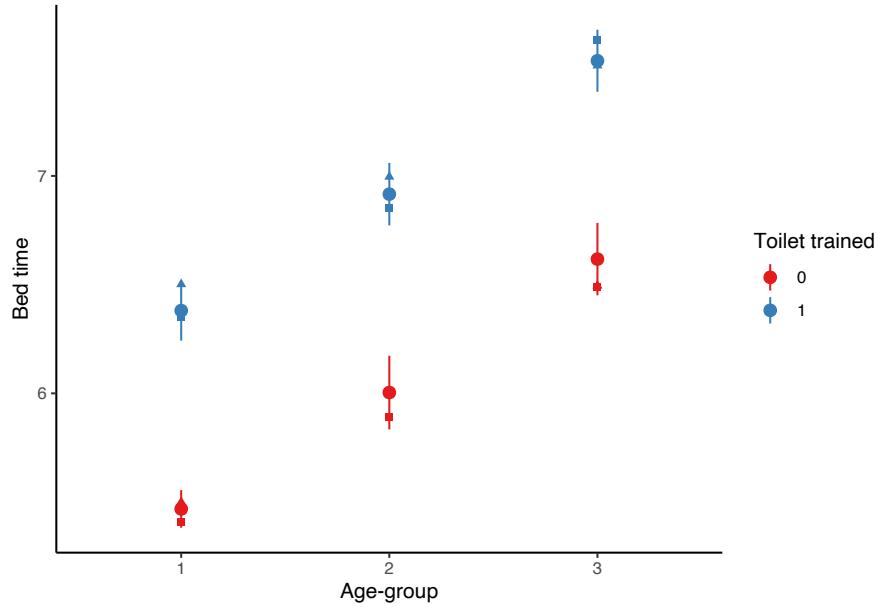
### 17.2.5 Post-stratify our model estimates

Now we create an estimate for each group, and add some confidence intervals.

```
poststratification_dataset_grouped <-
 mrp_example_model %>%
 predict(newdata = poststratification_dataset_grouped, interval = "confidence") %>%
 as_tibble() %>%
 cbind(poststratification_dataset_grouped, .) # The dot modifies the behaviour of the pipe; it pi
```

At this point we can have a look at our MRP estimates (circles) along with their confidence intervals, and compare them the raw estimates from the data (squares). In this case, because we know the truth, we can also compare them to the known truth (triangles) (but that's not something we can do normally).

```
poststratification_dataset_grouped %>%
 ggplot(aes(x = age_group, y = fit)) +
 geom_point(data = population_for_mrp_example_summarised,
 aes(x = age_group, y = median_bed_time, color = toilet_trained),
 shape = 17) +
 geom_point(data = sample_for_mrp_example_summarized,
 aes(x = age_group, y = median_bed_time, color = toilet_trained),
 shape = 15) +
 geom_pointrange(aes(ymin=lwr, ymax=upr, color = toilet_trained)) +
 labs(x = "Age-group",
 y = "Bed time",
 color = "Toilet trained") +
 theme_classic() +
 scale_color_brewer(palette = "Set1")
```



## 17.3 Case study - Xbox paper

### 17.3.1 Overview

One famous MRP example is [Wang et al. \(2015\)](#). They used data from the Xbox gaming platform to forecast the 2012 US Presidential Election.

Key facts about the set-up:

- Data are from an opt-in poll which was available on the Xbox gaming platform during the 45 days leading up to the 2012 US presidential election (Obama and Romney).
- Each day there were three to five questions, including voter intention: ‘If the election were held today, who would you vote for?’.
- Respondents were allowed to answer at most once per day.
- First-time respondents were asked to provide information about themselves, including their sex, race, age, education, state, party ID, political ideology, and who they voted for in the 2008 presidential election.
- In total, 750,148 interviews were conducted, with 345,858 unique respondents - over 30,000 of whom completed five or more polls.
- Young men dominate the Xbox population: 18-to-29-year-olds comprise 65 per cent of the Xbox dataset, compared to 19 per cent in the exit poll; and

men make up 93 per cent of the Xbox sample but only 47 per cent of the electorate.

### 17.3.2 Model

Given the structure of the US electorate, they use a two-stage modelling approach. The details don't really matter too much, but essentially they model how likely a respondent is to vote for Obama, given various information such as state, education, sex, etc:

$$Pr(Y_i = \text{Obama} | Y_i \in \{\text{Obama, Romney}\}) = \text{logit}^{-1}(\alpha_0 + \alpha_1(\text{state last vote share}) + \alpha_{j[i]}^{\text{state}} + \alpha_{j[i]}^{\text{edu}} + \alpha_{j[i]}^{\text{sex}} + \dots)$$

They run this in R using `glmer()` from 'lme4' (Bates et al., 2015).

### 17.3.3 Post-stratify

Having a trained model that considers the effect of these various independent variables on support for the candidates, they now post-stratify, where each of these 'cell-level estimates are weighted by the proportion of the electorate in each cell and aggregated to the appropriate level (i.e., state or national).'

This means that they need cross-tabulated population data. In general, the census would have worked, or one of the other large surveys available in the US, but the difficulty is that the variables need to be available on a cross-tab basis. As such, they use exit polls (not a viable option for most other countries).

They make state-specific estimates by post-stratifying to the features of each state (Figure 17.2).

Similarly, they can examine demographic-differences (Figure 17.3).

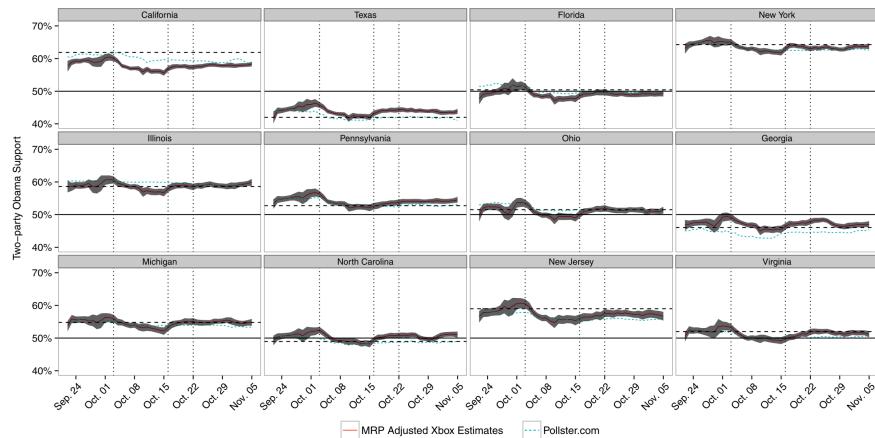
Finally, they convert their estimates into electoral college estimates (Figure 17.4).

## 17.4 Simulation - Australian voting

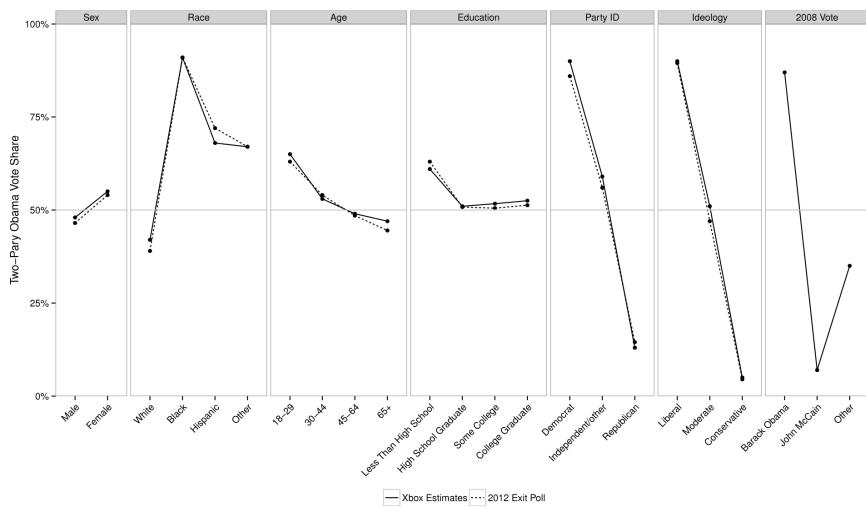
### 17.4.1 Overview

As a reminder, the workflow that we use is:

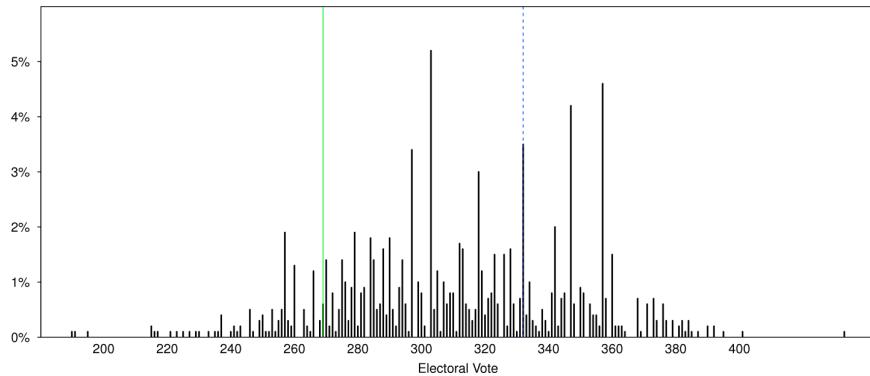
- 1) read in the poll;
- 2) model the poll;
- 3) read in the post-stratification data; and



**FIGURE 17.2:** Post-stratified estimates for each state based on the Xbox survey and MRP



**FIGURE 17.3:** Post-stratified estimates on a demographic basis based on the Xbox survey and MRP



**FIGURE 17.4:** Post-stratified estimates of electoral college outcomes based on the Xbox survey and MRP

- 4) apply the model to the post-stratification data.

In the earlier example, we didn't really do too much in the modelling step, and despite the name 'multilevel modelling with post-stratification', we didn't actually use a multilevel model. There's nothing that says you have to use a multilevel model, but a lot of situations will have circumstances such that it's not likely to do any worse. To be clear, this means that although we have individual-level data, there is some grouping of the individuals that we'll take advantage of. For instance, in the case of trying to model elections, usually districts/divisions/electorates/ridings/etc exist within provinces/states so it would likely make sense to, at least, include a coefficient that adjusts the intercept for each province.

In this section we're simulate another dataset and then fit a few different models to it. We're going to draw on the Australian elections set-up. In Australia we have a parliamentary system, with 151 seats in the parliament, one for each electorate. These electorates are grouped within six states and two territories. There are two major parties - the Australian Labor Party (ALP) and the Liberal Party (LP). Somewhat confusingly, the Liberal party are actually the conservative, right-wing party, while the Labor party are the progressive, left-wing, party.

#### 17.4.2 Construct a survey

To move us slightly closer to reality, we are going to simulate a survey (rather than sample from a population as we did earlier) and then post-stratify it using real data. The dependent variable is 'supports\_ALP', which is a binary variable - either 0 or 1. We'll just start with three independent variables here:

- ‘gender’, which is either ‘female’ or ‘male’ (as that is what is available from the Australian Bureau of Statistics);
- ‘age\_group’, which is one of four groups: ‘ages 18 to 29’, ‘ages 30 to 44’, ‘ages 45 to 59’, ‘ages 60 plus’;
- ‘state’, which is one of eight integers: 1 - 8 (inclusive).

At this point, it’s worth briefly discussing the role of sex and gender in survey research, following [Kennedy et al. \(2020\)](#). Sex is based on biological attributes, while gender is socially constructed. We are likely interested in the effect of gender on our dependent variable. Moving away from a non-binary concept of gender, in terms of official statistics, is only something that has happened recently. As a researcher one of the problems of insisting on a binary is that, as [Kennedy et al. \(2020, p. 2\)](#) say ‘...when measuring gender with simply two categories, there is a failure to capture the unique experiences of those who do not identify as either male or female, or for those whose gender does not align with their sex classification.’ A researcher has a variety of ways of proceeding, and [Kennedy et al. \(2020\)](#) discuss these based on: ethics, accuracy, practicality, and flexibility. However, ‘there is no single good solution that can be applied to all situations. Instead, it is important to recognize that there is a compromise between ethical concerns, statistical concerns, and the most appropriate decision will be reflective of this’ [p. 16]. The most important consideration is to ensure appropriate ‘respect and consideration for the survey respondent’.

```
library(tidyverse)
set.seed(853)

size_of_sample_for_australian_polling <- 2000

sample_for_australian_polling <-
 tibble(age_group =
 sample(x = c(0:3),
 size = size_of_sample_for_australian_polling,
 replace = TRUE
),
 gender =
 sample(x = c(0:1),
 size = size_of_sample_for_australian_polling,
 replace = TRUE
),
 state =
 sample(x = c(1:8),
 size = size_of_sample_for_australian_polling,
 replace = TRUE
),
```

```
noise = rnorm(size_of_sample_for_australian_polling, mean = 0, sd = 1),
support_alp = 1 + 0.5 * age_group + 0.5 * gender + 0.01 * state + noise
)

Normalize the outcome variable
sample_for_australian_polling <-
 sample_for_australian_polling %>%
 mutate(support_alp =
 if_else(support_alp > median(support_alp, na.rm = TRUE),
 'Supports ALP',
 'Does not')
)

Clean up the simulated data
sample_for_australian_polling <-
 sample_for_australian_polling %>%
 mutate(
 age_group = case_when(
 age_group == 0 ~ 'Ages 18 to 29',
 age_group == 1 ~ 'Ages 30 to 44',
 age_group == 2 ~ 'Ages 45 to 59',
 age_group == 3 ~ 'Ages 60 plus',
 TRUE ~ 'Problem'
),
 gender = case_when(
 gender == 0 ~ 'Male',
 gender == 1 ~ 'Female',
 TRUE ~ 'Problem'
),
 state = case_when(
 state == 1 ~ 'Queensland',
 state == 2 ~ 'New South Wales',
 state == 3 ~ 'Australian Capital Territory',
 state == 4 ~ 'Victoria',
 state == 5 ~ 'Tasmania',
 state == 6 ~ 'Northern Territory',
 state == 7 ~ 'South Australia',
 state == 8 ~ 'Western Australia',
 TRUE ~ 'Problem'
),
 ...
) %>%
 select(-noise)
```

```
Tidy the class
sample_for_australian_polling <-
 sample_for_australian_polling %>%
 mutate(across(c(age_group, gender, state, support_alp), as_factor))

sample_for_australian_polling %>%
 head()
#> # A tibble: 6 x 4
#> age_group gender state support_alp
#> <fct> <fct> <fct> <fct>
#> 1 Ages 18 to 29 Female South Australia Supports ALP
#> 2 Ages 60 plus Male South Australia Supports ALP
#> 3 Ages 30 to 44 Male Victoria Does not
#> 4 Ages 18 to 29 Male Tasmania Does not
#> 5 Ages 18 to 29 Female Victoria Does not
#> 6 Ages 18 to 29 Male Queensland Supports ALP
```

Finally, we want our survey to over-sample females, so we'll just get rid of 300 males.

```
sample_for_australian_polling <-
 sample_for_australian_polling %>%
 arrange(gender) %>%
 slice(1:1700)
```

### 17.4.3 Model the survey

This polling data was generated to make both males and older people less likely to vote for the ALP; and females and younger people more likely to vote for the Labor Party. Females are over-sampled. As such, we should have an ALP skew on the dataset. We're going to use the `gtsummary` package to quickly make a summary table (Sjoberg et al., 2021).

```
library(gtsummary)

sample_for_australian_polling %>%
 gtsummary::tbl_summary()
```

**Characteristic**	**N = 1,700**
age_group	
Ages 18 to 29	458 (27%)
Ages 60 plus	421 (25%)
Ages 30 to 44	401 (24%)
Ages 45 to 59	420 (25%)
gender	
Female	1,023 (60%)
Male	677 (40%)
state	
South Australia	233 (14%)
Victoria	189 (11%)
Tasmania	229 (13%)
Queensland	214 (13%)
Western Australia	198 (12%)
New South Wales	219 (13%)
Australian Capital Territory	237 (14%)
Northern Territory	181 (11%)
support_alp	
Supports ALP	896 (53%)
Does not	804 (47%)

Now we'd like to see if we can get our results back (we should find females less likely than males to vote for Australian Labor Party and that people are less likely to vote Australian Labor Party as they get older). Our model is:

#### ADD THE MODEL.

This model says that the probability that some person,  $j$ , will vote for the Australian Labor Party depends on their gender and their age-group. Based on our simulated data, we would like older age-groups to be less likely to vote for the Australian Labor Party and for males to be less likely to vote for the Australian Labor Party.

```
alp_support <-
 glm(support_alp ~ gender + age_group + state,
 data = sample_for_australian_polling,
 family = "binomial"
)

alp_support %>%
 modelsummary::modelsummary(fmt = 2, exponentiate = TRUE)
```

Essentially we've got our inputs back. Our dependent variable is a binary, and so we used logistic regression so the results are a little more difficult to interpret.

	Model 1
(Intercept)	1.44 (0.18)
genderMale	3.22 (0.12)
age_groupAges 60 plus	0.07 (0.17)
age_groupAges 30 to 44	0.42 (0.15)
age_groupAges 45 to 59	0.17 (0.15)
stateVictoria	1.65 (0.22)
stateTasmania	1.24 (0.21)
stateQueensland	1.46 (0.22)
stateWestern Australia	1.18 (0.22)
stateNew South Wales	1.42 (0.21)
stateAustralian Capital Territory	1.73 (0.21)
stateNorthern Territory	1.49 (0.23)
Num.Obs.	1700
AIC	1959.7
BIC	2024.9
Log.Lik.	-967.836
F	28.555

#### 17.4.4 Post-stratify

Now we'd like to see if we can use what we found in the poll to get an estimate for each state based on their demographic features.

First read in some real demographic data, on a state basis, from the ABS.

```
post_strat_census_data <-
 read_csv("outputs/data/census_data.csv")

head(post_strat_census_data)
#> # A tibble: 6 x 5
```

```
#> state gender age_group number cell_prop_of_division_total
#> <chr> <chr> <chr> <dbl> <dbl>
#> 1 ACT Female ages18to29 34683 0.125
#> 2 ACT Female ages30to44 42980 0.155
#> 3 ACT Female ages45to59 33769 0.122
#> 4 ACT Female ages60plus 30322 0.109
#> 5 ACT Male ages18to29 34163 0.123
#> 6 ACT Male ages30to44 41288 0.149
```

At this point, we've got a decision to make because we need the variables to be the same in the survey and the post-stratification dataset, but here the state abbreviations have been used, while in the survey, the full names were used. We'll change the post-stratification dataset because the survey data has already modelled.

```
post_strat_census_data <-
 post_strat_census_data %>%
 mutate(
 state =
 case_when(
 state == 'ACT' ~ 'Australian Capital Territory',
 state == 'NSW' ~ 'New South Wales',
 state == 'NT' ~ 'Northern Territory',
 state == 'QLD' ~ 'Queensland',
 state == 'SA' ~ 'South Australia',
 state == 'TAS' ~ 'Tasmania',
 state == 'VIC' ~ 'Victoria',
 state == 'WA' ~ 'Western Australia',
 TRUE ~ "Problem"
),
 age_group =
 case_when(
 age_group == 'ages18to29' ~ 'Ages 18 to 29',
 age_group == 'ages30to44' ~ 'Ages 30 to 44',
 age_group == 'ages45to59' ~ 'Ages 45 to 59',
 age_group == 'ages60plus' ~ 'Ages 60 plus',
 TRUE ~ "Problem"
)
)
```

We're just going to do some rough forecasts. For each gender and age-group we want the relevant coefficient in the example data and we can construct the estimates.

```

post_strat_census_data <-
 alp_support %>%
 predict(newdata = post_strat_census_data, type = 'response', se.fit = TRUE) %>%
 as_tibble() %>%
 cbind(post_strat_census_data, .)

post_strat_census_data %>%
 mutate(alp_predict_prop = fit*cell_prop_of_division_total) %>%
 group_by(state) %>%
 summarise(alp_predict = sum(alp_predict_prop))

#> # A tibble: 8 x 2
#> state alp_predict
#> <chr> <dbl>
#> 1 Australian Capital Territory 0.551
#> 2 New South Wales 0.487
#> 3 Northern Territory 0.546
#> 4 Queensland 0.491
#> 5 South Australia 0.403
#> 6 Tasmania 0.429
#> 7 Victoria 0.521
#> 8 Western Australia 0.460

```

We now have post-stratified estimates for each state. Our model has a fair few weaknesses. For instance, small cell counts are going to be problematic. And our approach ignores uncertainty, but now that we have something working we can complicate it.

#### 17.4.5 Improving the model

We'd like to address some of the major issues with our approach, specifically being able to deal with small cell counts, and also taking better account of uncertainty. As we are dealing with survey data, prediction intervals or something similar are critical, and it's not appropriate to only report central estimates. To do this we'll use the same broad approach as before, but just improve the model. We're going to change to a Bayesian model and use the `rstanarm` package (Goodrich et al., 2020).

Now, using the same basic model as before, but in a Bayesian setting.

```

library(rstanarm)

improved_alp_support <-
 rstanarm::stan_glm(support_alp ~ gender + age_group + state,
 data = sample_for_australian_polling,

```

```
family = binomial(link = "logit"),
prior = normal(0, 1),
prior_intercept = normal(0, 1),
cores = 2,
seed = 12345)
```

As before, we'd like an estimate for each state based on their demographic features. We're just going to do some rough forecasts. For each gender and age-group we want the relevant coefficient in the example data and we can construct the estimates (this code is from Monica Alexander<sup>4</sup>). We're going to use `tidybayes` for this (Kay, 2020).

```
library(tidybayes)

post_stratified_estimates <-
 improved_alp_support %>%
 tidybayes::add_fitted_draws(newdata = post_strat_census_data) %>%
 rename(alp_predict = .value) %>%
 mutate(alp_predict_prop = alp_predict * cell_prop_of_division_total) %>%
 group_by(state, .draw) %>%
 summarise(alp_predict = sum(alp_predict_prop)) %>%
 group_by(state) %>%
 summarise(mean = mean(alp_predict),
 lower = quantile(alp_predict, 0.025),
 upper = quantile(alp_predict, 0.975))

post_stratified_estimates
#> # A tibble: 8 x 4
#> state mean lower upper
#> <chr> <dbl> <dbl> <dbl>
#> 1 Australian Capital Territory 0.550 0.494 0.604
#> 2 New South Wales 0.486 0.429 0.544
#> 3 Northern Territory 0.544 0.483 0.607
#> 4 Queensland 0.491 0.432 0.548
#> 5 South Australia 0.412 0.361 0.464
#> 6 Tasmania 0.429 0.372 0.487
#> 7 Victoria 0.519 0.453 0.583
#> 8 Western Australia 0.460 0.401 0.520
```

We now have post-stratified estimates for each division. Our new Bayesian approach will enable us to think more deeply about uncertainty. We could

---

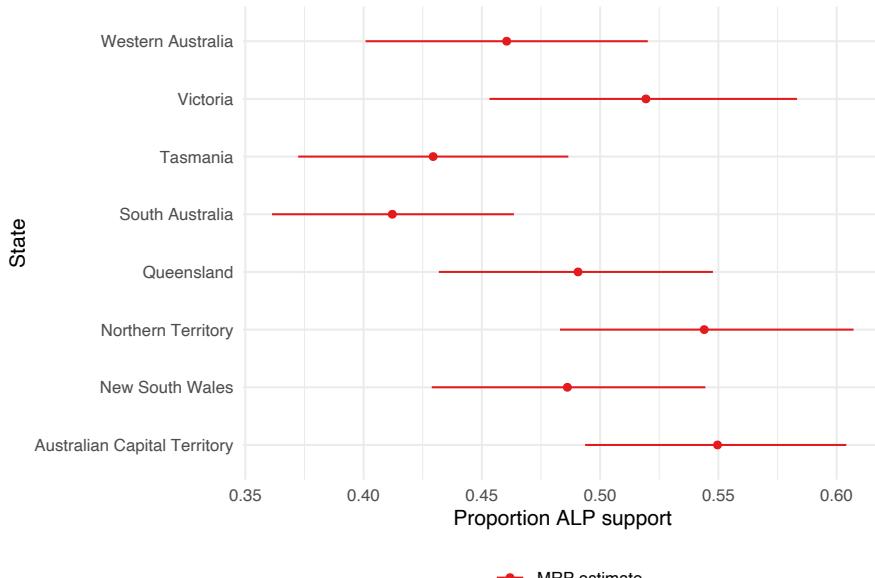
<sup>4</sup><https://www.monicaalexander.com/posts/2019-08-07-mrp/>

complicate this in a variety of ways including adding more coefficients (but remember that we'd need to get new cell counts), or adding some layers.

One interesting aspect is that our multilevel approach will allow us to deal with small cell counts by borrowing information from other cells. Even if we were to remove most of the, say, 18-to-29-year-old, male respondents from Tasmania our model would still provide estimates. It does this by pooling, in which the effect of these young, male, Tasmanians is partially determined by other cells that do have respondents.

There are many interesting aspects that we may like to communicate to others. For instance, we may like to show how the model is affecting the results. We can make a graph that compares the raw estimate with the model estimate.

```
post_stratified_estimates %>%
 ggplot(aes(y = mean, x = forcats::fct_inorder(state), color = "MRP estimate")) +
 geom_point() +
 geom_errorbar(aes(ymin = lower, ymax = upper), width = 0) +
 labs(y = "Proportion ALP support",
 x = "State") +
 theme_minimal() +
 scale_color_brewer(palette = "Set1") +
 theme(legend.position = "bottom") +
 theme(legend.title = element_blank()) +
 coord_flip()
```



Similarly, we may like to plot the distribution of the coefficients.<sup>5</sup>

```
tidybayes::get_variables(improved_alp_support)
improved_alp_support %>%
gather_draws(genderMale) %>%
ungroup() %>%
mutate(coefficient = stringr::str_replace_all(.variable, c("b_" = ""))) %>%
mutate(coefficient = forcats::fct_recode(coefficient,
Intercept = "Intercept",
`Is male` = "genderMale",
`Age 30-44` = "age_groupages30to44",
`Age 45-59` = "age_groupages45to59",
`Age 60+` = "age_groupages60plus"
)) %>%
#
both %>%
ggplot(aes(y=fct_rev(coefficient), x = .value)) +
ggridges::geom_density_ridges2(aes(height = ..density..),
rel_min_height = 0.01,
stat = "density",
scale=1.5) +
xlab("Distribution of estimate") +
ylab("Coefficient") +
scale_fill_brewer(name = "Dataset: ", palette = "Set1") +
theme_minimal() +
theme(panel.grid.major = element_blank(),
panel.grid.minor = element_blank()) +
theme(legend.position = "bottom")
```

## 17.5 Forecasting the 2020 US election

The US election has a lot of features that are unique to the US, but the model that we are going to build here is going to be fairly generic and, largely a generalization of the earlier model for the Australian election. One good thing about forecasting the US election is that there is a lot of data around. In this case we can use survey data from the Democracy Fund Voter Study

---

<sup>5</sup>You can work out which coefficients to be pass to `gather_draws` by using `tidybayes::get_variables(model)`. (In this example I passed '`b_`', but the ones of interest to you may be different.)

Group<sup>6</sup>.<sup>7</sup> They conducted polling in the lead-up to the US election and make this publicly available after registration. We will use the Integrated Public Use Microdata Series (IPUMS), to access the 2018 American Community Survey (ACS) as a post-stratification dataset. We will use state, age-group, gender, and education as explanatory variables.

### 17.5.1 Survey data

The first step is that we need to actually get the survey data. Go to their website:<https://www.voterstudygroup.org> and then you're looking for 'Nationscape' and then a button along the lines of 'Get the latest Nationscape data'. To get the dataset, you need to fill out a form, which they will process and then email you. There is a real person on the other side of this form, and so your request could take a few days.

Once you get access you'll want to download the .dta files. Nationscape conducted many surveys and so there are many files. The filename is the reference date, and so 'ns20200625' refers to 25 June 2020, which is the one that I'll use here. (This data is not mine to share, which is why I'll refer)

We can read in '.dta' files using the `haven` package (Wickham and Miller, 2020). I've based this code on that written by Alen Mitrovski, Xiaoyan Yang, Matthew Wankiewicz, which is available here: [https://github.com/matthewwankiewicz/US\\_election\\_forecast](https://github.com/matthewwankiewicz/US_election_forecast).

```
library(haven)
library(tidyverse)

raw_nationscape_data <-
 read_dta(here::here("dont_push/ns20200625.dta"))

The Stata format separates labels so reunite those
raw_nationscape_data <-
 labelled::to_factor(raw_nationscape_data)

Just keep relevant variables
nationscape_data <-
 raw_nationscape_data %>%
 select(vote_2020,
 gender,
 education,
 state,
 age)
```

---

<sup>6</sup><https://www.voterstudygroup.org>

<sup>7</sup>I thank Chris Warshaw<sup>8</sup> for putting me onto this dataset.

```

For simplicity, remove anyone undecided or planning to vote for someone other than Biden/Trump at
nationscape_data <-
 nationscape_data %>%
 filter(vote_2020 == "Joe Biden" | vote_2020 == "Donald Trump") %>%
 mutate(vote_biden = if_else(vote_2020 == "Joe Biden", 1, 0)) %>%
 select(-vote_2020)

Create the dependent variables by grouping the existing variables
nationscape_data <-
 nationscape_data %>%
 mutate(
 age_group = case_when(# case_when works in order and exits when there's a match
 age <= 29 ~ 'age_18-29',
 age <= 44 ~ 'age_30-44',
 age <= 59 ~ 'age_45-59',
 age >= 60 ~ 'age_60_or_more',
 TRUE ~ 'Trouble'
),
 gender = case_when(
 gender == "Female" ~ 'female',
 gender == "Male" ~ 'male',
 TRUE ~ 'Trouble'
),
 education_level = case_when(
 education == "3rd Grade or less" ~ "High school or less",
 education == "Middle School - Grades 4 - 8" ~ "High school or less",
 education == "Completed some high school" ~ "High school or less",
 education == "High school graduate" ~ "High school or less",
 education == "Other post high school vocational training" ~ "Some post secondary",
 education == "Completed some college, but no degree" ~ "Some post secondary",
 education == "Associate Degree" ~ "Post secondary or higher",
 education == "College Degree (such as B.A., B.S.)" ~ "Post secondary or higher",
 education == "Completed some graduate, but no degree" ~ "Post secondary or higher",
 education == "Masters degree" ~ "Graduate degree",
 education == "Doctorate degree" ~ "Graduate degree",
 TRUE ~ 'Trouble'
)
) %>%
 select(-education, -age)

tests <-
 nationscape_data %>%
 mutate(test = stringr::str_detect(age_group, 'Trouble'),

```

```

 test = if_else(test == TRUE, TRUE,
 stringr::str_detect(education_level, 'Trouble')),
 test = if_else(test == TRUE, TRUE,
 stringr::str_detect(gender, 'Trouble'))
) %>%
filter(test == TRUE)

if(nrow(tests) != 0) {
 print("Check nationscape_data")
} else {
 rm(tests)
}

nationscape_data %>%
 head()
#> # A tibble: 6 x 5
#> gender state vote_biden age_group education_level
#> <chr> <chr> <dbl> <chr> <chr>
#> 1 female WI 0 age_45-59 Post secondary or higher
#> 2 female VA 0 age_45-59 Post secondary or higher
#> 3 female TX 0 age_60_or_more High school or less
#> 4 female WA 0 age_45-59 High school or less
#> 5 female MA 1 age_18-29 Some post secondary
#> 6 female TX 1 age_30-44 Some post secondary

```

As we've seen, one of the most difficult aspects with MRP is ensuring consistency between the datasets. In this case, we need to do some work to make the variables consistent.

```

This code is very directly from Alen Mitrovski, Xiaoyan Yang, and Matthew Wankiewicz.
Format state names so the whole state name is written out, to match IPUMS data
states_names_and_abrevs <-
 tibble(stateicp = state.name, state = state.abb)

nationscape_data <-
 nationscape_data %>%
 left_join(states_names_and_abrevs)

rm(states_names_and_abrevs)

Make lowercase to match IPUMS data
nationscape_data <-
 nationscape_data %>%

```

```

 mutate(stateicp = tolower(stateicp))

Replace NAs with DC
nationscape_data$stateicp <-
 replace_na(nationscape_data$stateicp, "district of columbia")

Tidy the class
nationscape_data <-
 nationscape_data %>%
 mutate(across(c(gender, stateicp, education_level, age_group), as_factor))

Save data
write_csv(nationscape_data, "outputs/data/polling_data.csv")

nationscape_data %>%
 head()
#> # A tibble: 6 x 6
#> gender state vote_biden age_group education_level stateicp
#> <fct> <chr> <dbl> <fct> <fct> <fct>
#> 1 female WI 0 age_45-59 Post secondary or higher wisconsin
#> 2 female VA 0 age_45-59 Post secondary or higher virginia
#> 3 female TX 0 age_60_or_more High school or less texas
#> 4 female WA 0 age_45-59 High school or less washington
#> 5 female MA 1 age_18-29 Some post secondary massachusetts
#> 6 female TX 1 age_30-44 Some post secondary texas

```

### 17.5.2 Post-stratification data

We have a lot of options for a dataset to post-stratify by and there are various considerations. We are after a dataset that is better quality (however that is to be defined), and likely larger. From a strictly data perspective, the best choice would probably be something like the Cooperative Congressional Election Study (CCES), however for whatever reason that is only released after the election and so it's not a reasonable choice. Wang et al. (2015) use exit poll data, but again that's only available after the election.

In most countries we'd be stuck using the census, which is of course quite large, but likely out-of-date. Luckily in the US we have the opportunity to use the American Community Survey (ACS) which asks analogous questions to a census, is conducted every month, and over the course of a year, we end up with a few million responses. In this case we're going to access the ACS through IPUMS.

To do this go to the IPUMS website - <https://ipums.org> - and we're looking

for something like IPUMS USA and then ‘get data’. Create an account, if you need to. That’ll take a while to process. But once you have an account, go to ‘Select Samples’ and de-select everything apart from the 2019 ACS. Then we need to get the variables that we’re interested in. From the household we want ‘STATEICP’, then in person we want ‘SEX’, ‘AGE’, ‘EDUC’. Once everything is selected, ‘view cart’, and we want to be careful to change the ‘data format’ to ‘.dta’ (there’s nothing wrong with the other formats, but we’ve just already got code earlier to deal with that type). Briefly just check how many rows and columns you’re requesting. It should be around a million rows, and around ten to twenty columns. If it’s much more than 300MB then maybe just see if you’ve accidentally selected something that you don’t need. Submit the request and within a day, you should get an email saying that your data can be downloaded. It should only take 30 minutes or so, but if you don’t get an email within a day then check again the size of the dataset, and customize the sample size to reduce the size initially.

In any case let’s tidy up the data.

```
Again, closely following code from Alen Mitrovski, Xiaoyan Yang, and Matthew Wankiewicz.
library(haven)
library(tidyverse)

raw_poststrat_data <-
 read_dta(here::here("dont_push/usa_00004.dta"))

The Stata format separates labels so reunite those
raw_poststrat_data <-
 labelled::to_factor(raw_poststrat_data)
head(raw_poststrat_data)
#> # A tibble: 6 x 28
#> year sample serial cbserial hhwt cluster region stateicp strata gq pernum
#> <fct> <fct> <dbl> <dbl> <dbl> <dbl> <fct> <fct> <dbl> <fct> <dbl>
#> 1 2018 2018 ~ 2 2.02e12 392. 2.02e12 east ~ alabama 190001 othe~ 1
#> 2 2018 2018 ~ 7 2.02e12 94.1 2.02e12 east ~ alabama 400001 grou~ 1
#> 3 2018 2018 ~ 13 2.02e12 83.7 2.02e12 east ~ alabama 130301 othe~ 1
#> 4 2018 2018 ~ 18 2.02e12 57.5 2.02e12 east ~ alabama 100001 grou~ 1
#> 5 2018 2018 ~ 23 2.02e12 157. 2.02e12 east ~ alabama 190001 grou~ 1
#> 6 2018 2018 ~ 28 2.02e12 157. 2.02e12 east ~ alabama 220001 othe~ 1
#> # ... with 17 more variables: perwt <dbl>, sex <fct>, age <fct>, marst <fct>,
#> # race <fct>, raced <fct>, hispan <fct>, hispand <fct>, bpl <fct>,
#> # bpld <fct>, citizen <fct>, educ <fct>, educcd <fct>, empstat <fct>,
#> # empstata <fct>, labforce <fct>, inctot <dbl>

raw_poststrat_data$age <- as.numeric(raw_poststrat_data$age)
```

```

poststrat_data <-
 raw_poststrat_data %>%
 filter(inctot < 9999999) %>%
 filter(age >= 18) %>%
 mutate(gender = sex) %>%
 mutate(
 age_group = case_when(# case_when works in order and exits when there's a match
 age <= 29 ~ 'age_18-29',
 age <= 44 ~ 'age_30-44',
 age <= 59 ~ 'age_45-59',
 age >= 60 ~ 'age_60_or_more',
 TRUE ~ 'Trouble'
),
 education_level = case_when(
 educd == "nursery school, preschool" ~ "High school or less",
 educd == "kindergarten" ~ "High school or less",
 educd == "grade 1" ~ "High school or less",
 educd == "grade 2" ~ "High school or less",
 educd == "grade 3" ~ "High school or less",
 educd == "grade 4" ~ "High school or less",
 educd == "grade 5" ~ "High school or less",
 educd == "grade 6" ~ "High school or less",
 educd == "grade 7" ~ "High school or less",
 educd == "grade 8" ~ "High school or less",
 educd == "grade 9" ~ "High school or less",
 educd == "grade 10" ~ "High school or less",
 educd == "grade 11" ~ "High school or less",
 educd == "12th grade, no diploma" ~ "High school or less",
 educd == "regular high school diploma" ~ "High school or less",
 educd == "ged or alternative credential" ~ "High school or less",
 educd == "some college, but less than 1 year" ~ "Some post secondary",
 educd == "1 or more years of college credit, no degree" ~ "Some post secondary",
 educd == "associate's degree, type not specified" ~ "Post secondary or higher",
 educd == "bachelor's degree" ~ "Post secondary or higher",
 educd == "master's degree" ~ "Graduate degree",
 educd == "professional degree beyond a bachelor's degree" ~ "Graduate degree",
 educd == "doctoral degree" ~ "Graduate degree",
 educd == "no schooling completed" ~ "High school or less",
 TRUE ~ 'Trouble'
)
)

Just keep relevant variables
poststrat_data <-

```

```

poststrat_data %>%
 select(gender,
 age_group,
 education_level,
 stateicp)

Tidy the class
poststrat_data <-
 poststrat_data %>%
 mutate(across(c(gender, stateicp, education_level, age_group), as_factor))

Save data
write_csv(poststrat_data, "outputs/data/us_poststrat.csv")

poststrat_data %>%
 head()
#> # A tibble: 6 x 4
#> gender age_group education_level stateicp
#> <fct> <fct> <fct> <fct>
#> 1 female age_18-29 Some post secondary alabama
#> 2 female age_60_or_more Some post secondary alabama
#> 3 male age_45-59 Some post secondary alabama
#> 4 male age_30-44 High school or less alabama
#> 5 female age_60_or_more High school or less alabama
#> 6 male age_30-44 High school or less alabama

```

This dataset is on an individual level. So we'll create counts of each sub-cell, and then proportions by state.

```

poststrat_data_cells <-
 poststrat_data %>%
 group_by(stateicp, gender, age_group, education_level) %>%
 count()

```

Now we'd like to add proportions by state.

```

poststrat_data_cells <-
 poststrat_data_cells %>%
 group_by(stateicp) %>%
 mutate(prop = n/sum(n)) %>%
 ungroup()

poststrat_data_cells %>% head()

```

```
#> # A tibble: 6 x 6
#> stateicp gender age_group education_level n prop
#> <fct> <fct> <fct> <fct> <int> <dbl>
#> 1 connecticut male age_18-29 Some post secondary 149 0.0260
#> 2 connecticut male age_18-29 High school or less 232 0.0404
#> 3 connecticut male age_18-29 Post secondary or higher 96 0.0167
#> 4 connecticut male age_18-29 Graduate degree 25 0.00436
#> 5 connecticut male age_60_or_more Some post secondary 142 0.0248
#> 6 connecticut male age_60_or_more High school or less 371 0.0647
```

### 17.5.3 Model

We're going to use logistic regression to estimate a model where the binary of support for Biden is explained by gender, age-group, education-level, and state. We're going to do this in a Bayesian framework using `rstanarm` (Goodrich et al., 2020). There are a variety of reasons for using `rstanarm` here, but the main one is that Stan is pre-compiled which eases some of the computer set-up issues that we may otherwise have. A great further resource about implementing MRP with `rstanarm` is Kennedy and Gabry (2020).

```
library(rstanarm)

us_election_model <-
 rstanarm::stan_glmer(vote_biden ~ gender + age_group + (1 | stateicp) + education_level,
 data = nationscape_data,
 family = binomial(link = "logit"),
 prior = normal(0, 1),
 prior_intercept = normal(0, 1),
 cores = 2,
 seed = 853)
```

There are a variety of options here that we've largely unthinkingly set, and exploring the effect of these would be a good idea, but for now we can just have a quick look at the model.

```
modelsummary::get_estimates(us_election_model)
#> term effect estimate conf.level
#> 1 (Intercept) fixed 0.27591651 0.95
#> 2 gendermale fixed -0.54094841 0.95
#> 3 age_groupage_60_or_more fixed 0.04886803 0.95
#> 4 age_groupage_18-29 fixed 0.87817445 0.95
#> 5 age_groupage_30-44 fixed 0.12455081 0.95
```

```

#> 6 education_levelHigh school or less fixed -0.35080948 0.95
#> 7 education_levelsome post secondary fixed -0.14970941 0.95
#> 8 education_levelGraduate degree fixed -0.21988079 0.95
#> 9 Sigma[stateicp:(Intercept),(Intercept)] random 0.08002654 0.95
#> conf.low conf.high pd rope.percentage rhat ess
#> 1 0.10322566 0.447878764 0.99850 0.1281242 1.0002399 2262.704
#> 2 -0.65227570 -0.430121742 1.00000 0.0000000 0.9996776 5264.121
#> 3 -0.10847716 0.200831286 0.72600 0.9765851 0.9998468 3726.472
#> 4 0.69830955 1.061899702 1.00000 0.0000000 0.9997188 3931.709
#> 5 -0.02530517 0.284994549 0.94125 0.7729545 0.9995061 3567.152
#> 6 -0.51262507 -0.204222132 1.00000 0.0000000 0.9997706 3700.321
#> 7 -0.29927232 0.006135831 0.96625 0.6771902 1.0005471 4090.628
#> 8 -0.38172115 -0.036764485 0.99100 0.3249145 0.9999286 4589.040
#> 9 0.02912381 0.160211514 1.00000 1.0000000 1.0031942 1284.436
#> prior.distribution prior.location prior.scale
#> 1 normal 0 1
#> 2 normal 0 1
#> 3 normal 0 1
#> 4 normal 0 1
#> 5 normal 0 1
#> 6 normal 0 1
#> 7 normal 0 1
#> 8 normal 0 1
#> 9 <NA> NA NA
The default usage of modelsummary requires statistics that we don't have.
Uncomment the following line if you want to look at what is available and specify your own:
modelsummary::get_estimates(us_election_model)
modelsummary::modelsummary(us_election_model,
 statistic = c('conf.low', 'conf.high')
)

```

#### 17.5.4 Post-stratify

```

biden_support_by_state <-
 us_election_model %>%
 tidybayes::add_fitted_draws(newdata=poststrat_data_cells) %>%
 rename(support_biden_predict = .value) %>%
 mutate(support_biden_predict_prop = support_biden_predict*prop) %>%
 group_by(stateicp, .draw) %>%
 summarise(support_biden_predict = sum(support_biden_predict_prop)) %>%
 group_by(stateicp) %>%
 summarise(mean = mean(support_biden_predict),

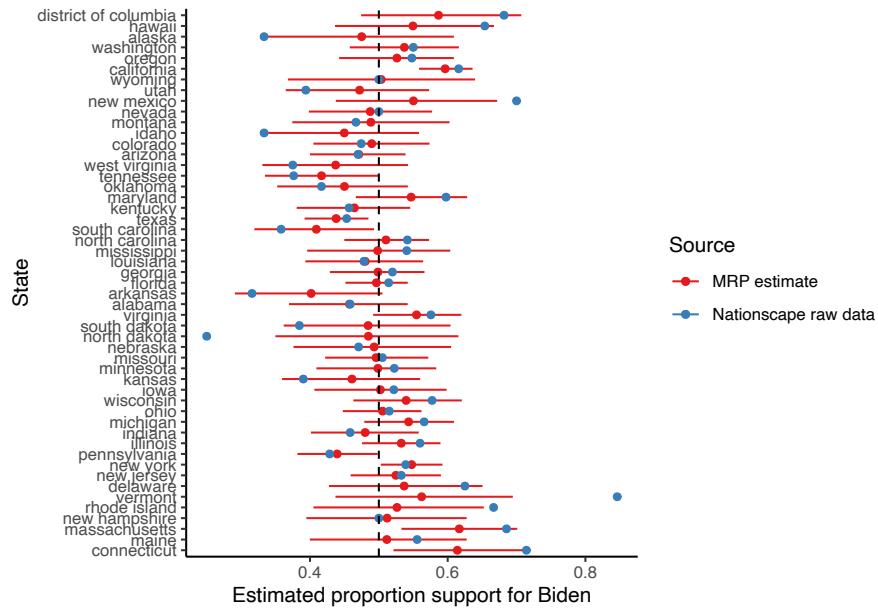
```

	Model 1
(Intercept)	0.276 (0.103) (0.448)
gendermale	-0.541 (-0.652) (-0.430)
age_groupage_60_or_more	0.049 (-0.108) (0.201)
age_groupage_18-29	0.878 (0.698) (1.062)
age_groupage_30-44	0.125 (-0.025) (0.285)
education_levelHigh school or less	-0.351 (-0.513) (-0.204)
education_levelSome post secondary	-0.150 (-0.299) (0.006)
education_levelGraduate degree	-0.220 (-0.382) (-0.037)
Sigma[stateicp × (Intercept),(Intercept)]	0.080 (0.029) (0.160)
Num.Obs.	5200
R2	0.057
R2 Marg.	0.045
ELPD	-3468.0
ELPD s.e.	16.4
LOOIC	6936.1
LOOIC s.e.	32.8
WAIC	6936.0
RMSE	0.48

```
lower = quantile(support_biden_predict, 0.025),
upper = quantile(support_biden_predict, 0.975))
```

And we can have a look at our estimates, if we like.

```
biden_support_by_state %>%
 ggplot(aes(y = mean, x = stateicp, color = "MRP estimate")) +
 geom_point() +
 geom_errorbar(aes(ymin = lower, ymax = upper), width = 0) +
 geom_point(data =
 nationscape_data %>%
 group_by(stateicp, vote_biden) %>%
 summarise(n = n()) %>%
 group_by(stateicp) %>%
 mutate(prop = n/sum(n)) %>%
 filter(vote_biden==1),
 aes(y = prop, x = stateicp, color = 'Nationscape raw data')) +
 geom_hline(yintercept = 0.5, linetype = 'dashed') +
 labs(x = 'State',
 y = 'Estimated proportion support for Biden',
 color = 'Source') +
 theme_classic() +
 scale_color_brewer(palette = 'Set1') +
 coord_flip()
```



## 17.6 Concluding remarks and next steps

In general, MRP is a good way to accomplish specific aims, but it's not without trade-offs. If you have a good quality survey, then it may be a way to speak to disaggregated aspects of it. Or if you are concerned about uncertainty then it is a good way to think about that. If you have a biased survey then it's a great place to start, but it's not a panacea.

There's not a lot of work that's been done with it, so there's plenty of scope for exciting work from a variety of approaches:

- From a more statistical perspective, there is a lot of work to do in terms of thinking through how survey design and modelling approaches interact and the extent to which we are underestimating uncertainty. I'm also very interested in thinking through the implications of small samples and uncertainty in the post-stratification dataset. There's an awful lot to do in terms of thinking through what the appropriate model is to use, and how do we even evaluate what 'appropriate' means here? Those with statistical interests, should probably go next to Gao et al. (2021) as well as pretty much anything by Yajuan Si<sup>9</sup>, but Si (2020) would be a starting point.
- There's a lot to be done from a sociology perspective in terms of survey

<sup>9</sup><http://www-personal.umich.edu/~yajuan/>

responses and how we can better design our surveys, knowing they are going to be used for MRP and putting respect for our respondents first.

- From a political science perspective, we just have very little idea of the conditions under which we will have the stable preferences and relationships that are required for MRP to be accurate, and further understanding how this relates to uncertainty in survey design. For those with political science interests, a natural next step would be to go through [Lauderdale et al. \(2020\)](#) or [Ghitza and Gelman \(2020\)](#).
- Economists might be interested to think about how we could use MRP to better understand the inflation and unemployment rates at local levels.
- From a statistical software side of things, we really need to develop better packages around this.
- It's from an information side of things, that I'm most excited about MRP. How do we best store and protect our datasets, yet retain the ability to have them correspond with each other? How do we put the levels together in a way that is meaningful? To what extent do people appreciate uncertainty estimates and how can we better communicate these estimates?

More generally, we could pretty much use MRP anywhere we have samples. Determining the conditions under which we actually should, is the work of whole generations.

---

## 17.7 Exercises and tutorial

### 17.7.1 Exercises

### 17.7.2 Tutorial



# 18

---

## *Text as data*

---

**STATUS:** Under construction.

### Required reading

- Hvitfeldt, Emil, and Julia Silge, 2021, *Supervised Machine Learning for Text Analysis in R*, Chapters 2, 5, 6, 7, <https://smltar.com>.
- Silge, Julia, and David Robinson, 2017, *Text Mining with R*, <https://www.tidytextmining.com>.

### Required viewing

- 

### Recommended reading

- Amaka, Ofunne, and Amber Thomas, 2020, ‘The Naked Truth: How the names of 6,816 complexion products can reveal bias in beauty’, The Pudding, March, <https://pudding.cool/2021/03/foundation-names/>.

### Key concepts/skills/etc

- 

### Key libraries

- 

### Key functions/etc

- 

### Quiz

- 

---

### 18.1 Introduction

Text can be thought of as an unwieldy, but in generally

## 18.2 Lasso regression

This subsection, and much of the code that is used, directly draws on Julia Silge's notes, in particular: <https://juliasilge.com/blog/tidy-text-classification/> (Silge, 2018).

One of the nice aspects of text is that we can adapt our existing methods to use it as an input. Here we are going to use a variation of logistics regression, along with text inputs, to forecast. If you want to learn more about Lasso regression, then you should consider taking Arik's course over the summer, where he will dive into machine learning using Python.

In this section we are going to have two different text inputs, train a model on a sample of text from each of them, and then try to use that model to forecast the text in a training set. Although this is a arbitrary example, you could imagine many real-world applications. For instance, if you work at Twitter then you may want to know if a tweet was likely written by a bot, or by a human. Or similarly, imagine that you work for a political party - you may like to know if an email was likely from an email campaign organised by a group, or from an individual.

First we need to get some data. Julia Silge's example, nicely, uses book text as input. Seeing as I am jointly appointed at a Faculty of Information, that seems especially nice. The wonderful thing about this is that there is an R package - `gutenbergr` - that makes it easy to get text from Project Gutenberg into R. The key function is `gutenberg_download()`, which needs a key for the book that you want. We'll consider Jane Eyre and Alice's Adventures in Wonderland, which have the keys of 1260 and 11, respectively.

```
library(gutenbergr)
alice_and_jane <- gutenbergr::gutenberg_download(c(1260, 11), meta_fields = "title")

Save the dataset so that we don't need to overwhelm the servers each time
write_csv(alice_and_jane, "inputs/books/alice_and_jane.csv")

head(alice_and_jane)
```

```
library(gutenbergr)

alice_and_jane <- read_csv("inputs/books/alice_and_jane.csv")

head(alice_and_jane)
#> # A tibble: 6 x 3
```

One of the great things about this is that the dataset is a tibble. So we can just work with all our familiar skills. The package has a lot more functionality, so I'd encourage you to look at the package's website: <https://github.com/ropensci/gutenbergr>. Each line of the book is read in as a different row in the dataset. Notice that we have downloaded two books here at once, and so we added the title. The two books are one after each other. You can see that we have both by looking at some summary statistics.

So it looks like Jane Eyre is much longer than Alice in Wonderland, which isn't a surprise to those who have read them. I don't want to step into Digital Humanities too much, as I don't know anything about it, but looking at things like the broader context of when these books were written, or other books that were written at similar times, is likely a fascinating area.

We'll just get rid of blank lines

```
library(janitor)
TODO There's a way to do this within janitor, but I forget, need to look it up.
alice_and_jane <-
 alice_and_jane %>%
 mutate(blank_line = if_else(text == "", 1, 0)) %>%
 filter(blank_line == 0) %>%
 select(-blank_line)

table(alice_and_jane$title)
#>
#> Alice's Adventures in Wonderland Jane Eyre: An Autobiography
#> 2481 16395
```

There's still an overwhelming amount of Jane Eyre in there. So we'll just sample from Jane Eyre to make it more equal.

```
set.seed(853)

alice_and_jane$rows <- c(1:nrow(alice_and_jane))
sample_from_me <- alice_and_jane %>% filter(title == "Jane Eyre: An Autobiography")
keep_me <- sample(x = sample_from_me$rows, size = 2481, replace = FALSE)

alice_and_jane <-
 alice_and_jane %>%
 filter(title == "Alice's Adventures in Wonderland" | rows %in% keep_me) %>%
 select(-rows)

table(alice_and_jane$title)
#>
#> Alice's Adventures in Wonderland Jane Eyre: An Autobiography
#> 2481 2481
```

There's a bunch of issues here, for instance, we have the whole of Alice, but we only have random bits of Jane, but nonetheless let's continue and we'll try to do something about that in a moment.

Now we want to get a sample of text from each book. We will use the lines to distinguish these samples. So we use a counter that will add a line number.

```
alice_and_jane <-
 alice_and_jane %>%
 group_by(title) %>%
 mutate(line_number = paste(gutenberg_id, row_number(), sep = "_")) %>%
 ungroup()
```

We now want to separate out the words. We'll just use tidytext, because the focus here is on modelling, but there are a bunch of alternatives and one especially good one is the quanteda package, specifically, the tokens() function.

```
library(tidytext)
alice_and_jane_by_word <-
 alice_and_jane %>%
 unnest_tokens(word, text) %>%
 group_by(word) %>%
 filter(n() > 10) %>%
 ungroup()
```

Notice here that we removed any word that wasn't used more than 10 times. Nonetheless we still have a lot of unique words. (If we didn't require that the word be used by the author at least 10 times then we end up with more than 6,000 words.)

```
alice_and_jane_by_word$word %>% unique() %>% length()
#> [1] 585
```

The reason this is relevant is because these are our independent variables. So where you may be used to having something less than 10 explanatory variables, in this case we are going to have 585 As such, we need a model that can handle this.

However, as mentioned before, we are going to have some rows that essentially just had one word. While we could allow that, it might also be nice to give the model at least a few words to work with.

```
alice_and_jane_by_word <-
 alice_and_jane_by_word %>%
 group_by(title, line_number) %>%
 mutate(number_of_words_in_line = n()) %>%
 ungroup() %>%
 filter(number_of_words_in_line > 2) %>%
 select(-number_of_words_in_line)
```

We'll create a test/training split, and load in `tidymodels`.

```
library(tidymodels)

set.seed(853)

alice_and_jane_by_word_split <-
 alice_and_jane_by_word %>%
 select(title, line_number) %>%
 distinct() %>%
 initial_split(prop = 3/4, strata = title)

alice_and_jane_by_word_train <- training(alice_and_jane_by_word_split) %>% select(line_number)
alice_and_jane_by_word_test <- testing(alice_and_jane_by_word_split)
#
rm(alice_and_jane_by_word_split)
```

Now we need to create a document-term matrix.

```
alice_and_jane_dtm_training <-
 alice_and_jane_by_word %>%
 count(line_number, word) %>%
 inner_join(training(alice_and_jane_by_word_split) %>% select(line_number)) %>%
 cast_dtm(term = word, document = line_number, value = n)

dim(alice_and_jane_dtm_training)
#> [1] 3413 585
```

So we have our independent variables sorted, now we need our binary dependent variable, which is whether the book is Alice in Wonderland or Jane Eyre.

```
response <-
 data.frame(id = dimnames(alice_and_jane_dtm_training)[[1]]) %>%
 separate(id, into = c("book", "line", sep = "_")) %>%
 mutate(is_alice = if_else(book == 11, 1, 0))

predictor <- alice_and_jane_dtm_training[] %>% as.matrix()
```

Now we can run our model.

```
library(glmnet)

model <- cv.glmnet(x = predictor,
 y = response$is_alice,
 family = "binomial",
 keep = TRUE
)

save(model, file = "outputs/models/alice_vs_jane.rda")
```

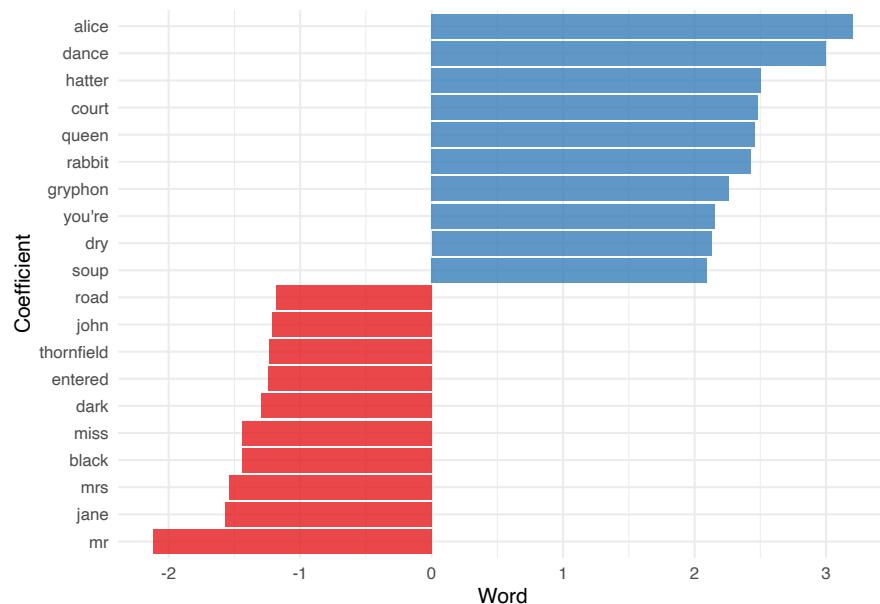
```
load("outputs/models/alice_vs_jane.rda")
library(glmnet)
library(broom)

coefs <- model$glmnet.fit %>%
 tidy() %>%
 filter(lambda == model$lambda.1se)

coefs %>% head()
```

```
#> # A tibble: 6 x 5
#> term step estimate lambda dev.ratio
#> <chr> <dbl> <dbl> <dbl> <dbl>
#> 1 (Intercept) 36 -0.335 0.00597 0.562
#> 2 in 36 -0.144 0.00597 0.562
#> 3 she 36 0.390 0.00597 0.562
#> 4 so 36 0.00249 0.00597 0.562
#> 5 a 36 -0.117 0.00597 0.562
#> 6 about 36 0.279 0.00597 0.562
```

```
coefs %>%
 group_by(estimate > 0) %>%
 top_n(10, abs(estimate)) %>%
 ungroup() %>%
 ggplot(aes(fct_reorder(term, estimate), estimate, fill = estimate > 0)) +
 geom_col(alpha = 0.8, show.legend = FALSE) +
 coord_flip() +
 theme_minimal() +
 labs(x = "Coefficient",
 y = "Word") +
 scale_fill_brewer(palette = "Set1")
```



Perhaps unsurprisingly, if you mention Alice then it's likely to be a Alice in Wonderland and if you mention Jane then it's likely to be Jane Eyre.

---

### 18.3 Topic models

A version of these notes was previously circulated as part of [Alexander and Alexander \(2021\)](#).

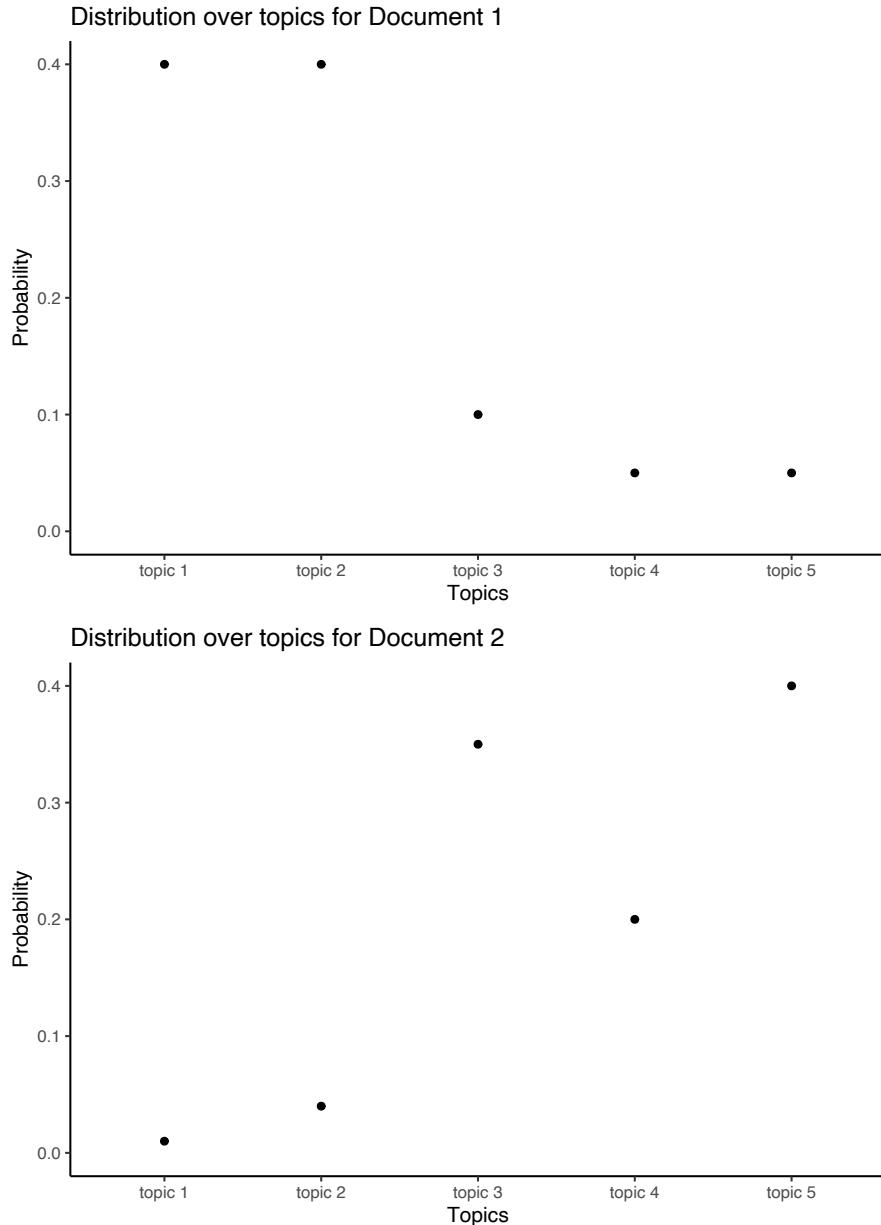
#### 18.3.1 Overview

Sometimes we have a statement and we want to know what it is about. Sometimes this will be easy, but we don't always have titles for statements, and even when we do, sometimes we do not have titles that define topics in a well-defined and consistent way. One way to get consistent estimates of the topics of each statement is to use topic models. While there are many variants, one way is to use the latent Dirichlet allocation (LDA) method of [Blei et al. \(2003\)](#), as implemented by the R package ‘topicmodels’ by [Grün and Hornik \(2011\)](#).

The key assumption behind the LDA method is that each statement, ‘a document’, is made by a person who decides the topics they would like to talk about in that document, and then chooses words, ‘terms’, that are appropriate to those topics. A topic could be thought of as a collection of terms, and a document as a collection of topics. The topics are not specified *ex ante*; they are an outcome of the method. Terms are not necessarily unique to a particular topic, and a document could be about more than one topic. This provides more flexibility than other approaches such as a strict word count method. The goal is to have the words found in documents group themselves to define topics.

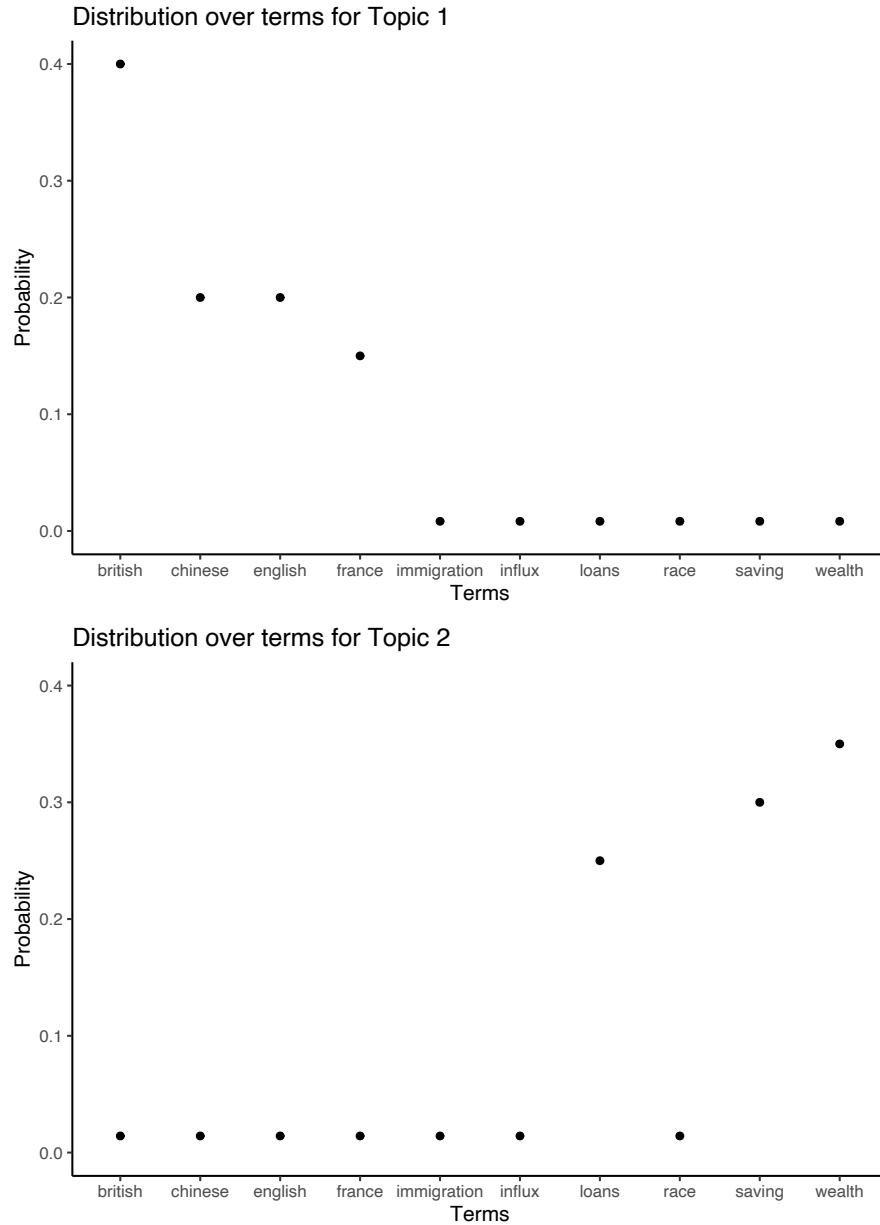
#### 18.3.2 Document generation process

The LDA method considers each statement to be a result of a process where a person first chooses the topics they want to speak about. After choosing the topics, the person then chooses appropriate words to use for each of those topics. More generally, the LDA topic model works by considering each document as having been generated by some probability distribution over topics. For instance, if there were five topics and two documents, then the first document may be comprised mostly of the first few topics; the other document may be mostly about the final few topics (Figure ??).



Similarly, each topic could be considered a probability distribution over terms. To choose the terms used in each document the speaker picks terms from each topic in the appropriate proportion. For instance, if there were ten terms, then one topic could be defined by giving more weight to terms related to

immigration; and some other topic may give more weight to terms related to the economy (Figure ??).



Following Blei and Lafferty (2009), Blei (2012) and Griffiths and Steyvers (2004), the process by which a document is generated is more formally considered to be:

1. There are  $1, 2, \dots, k, \dots, K$  topics and the vocabulary consists of  $1, 2, \dots, V$  terms. For each topic, decide the terms that the topic uses by randomly drawing distributions over the terms. The distribution over the terms for the  $k$ th topic is  $\beta_k$ . Typically a topic would be a small number of terms and so the Dirichlet distribution with hyperparameter  $0 < \eta < 1$  is used:  $\beta_k \sim \text{Dirichlet}(\eta)$ .<sup>1</sup> Strictly,  $\eta$  is actually a vector of hyperparameters, one for each  $K$ , but in practice they all tend to be the same value.
2. Decide the topics that each document will cover by randomly drawing distributions over the  $K$  topics for each of the  $1, 2, \dots, d, \dots, D$  documents. The topic distributions for the  $d$ th document are  $\theta_d$ , and  $\theta_{d,k}$  is the topic distribution for topic  $k$  in document  $d$ . Again, the Dirichlet distribution with the hyperparameter  $0 < \alpha < 1$  is used here because usually a document would only cover a handful of topics:  $\theta_d \sim \text{Dirichlet}(\alpha)$ . Again, strictly  $\alpha$  is vector of length  $K$  of hyperparameters, but in practice each is usually the same value.
3. If there are  $1, 2, \dots, n, \dots, N$  terms in the  $d$ th document, then to choose the  $n$ th term,  $w_{d,n}$ :
  - a. Randomly choose a topic for that term  $n$ , in that document  $d$ ,  $z_{d,n}$ , from the multinomial distribution over topics in that document,  $z_{d,n} \sim \text{Multinomial}(\theta_d)$ .
  - b. Randomly choose a term from the relevant multinomial distribution over the terms for that topic,  $w_{d,n} \sim \text{Multinomial}(\beta_{z_{d,n}})$ .

Given this set-up, the joint distribution for the variables is (Blei (2012), p.6):

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D,1:N}, w_{1:D,1:N}) = \prod_{i=1}^K p(\beta_i) \prod_{d=1}^D p(\theta_d) \left( \prod_{n=1}^N p(z_{d,n}|\theta_d) p(w_{d,n}|\beta_{1:K}, z_{d,n}) \right).$$

Based on this document generation process the analysis problem, discussed in the next section, is to compute a posterior over  $\beta_{1:K}$  and  $\theta_{1:D}$ , given  $w_{1:D,1:N}$ . This is intractable directly, but can be approximated (Griffiths and Steyvers (2004) and Blei (2012)).

### 18.3.3 Analysis process

After the documents are created, they are all that we have to analyse. The term usage in each document,  $w_{1:D,1:N}$ , is observed, but the topics are hidden, or ‘latent’. We do not know the topics of each document, nor how terms defined the topics. That is, we do not know the probability distributions of Figures

---

<sup>1</sup>The Dirichlet distribution is a variation of the beta distribution that is commonly used as a prior for categorical and multinomial variables. If there are just two categories, then the Dirichlet and the beta distributions are the same. In the special case of a symmetric Dirichlet distribution,  $\eta = 1$ , it is equivalent to a uniform distribution. If  $\eta < 1$ , then the distribution is sparse and concentrated on a smaller number of the values, and this number decreases as  $\eta$  decreases. A hyperparameter is a parameter of a prior distribution.

?? or ???. In a sense we are trying to reverse the document generation process – we have the terms and we would like to discover the topics.

If the earlier process around how the documents were generated is assumed and we observe the terms in each document, then we can obtain estimates of the topics ([Steyvers and Griffiths \(2006\)](#)). The outcomes of the LDA process are probability distributions and these define the topics. Each term will be given a probability of being a member of a particular topic, and each document will be given a probability of being about a particular topic. That is, we are trying to calculate the posterior distribution of the topics given the terms observed in each document ([Blei \(2012\)](#), p.7):

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D,1:N} | w_{1:D,1:N}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D,1:N}, w_{1:D,1:N})}{p(w_{1:D,1:N})}.$$

The initial practical step when implementing LDA given a corpus of documents is to remove ‘stop words’. These are words that are common, but that don’t typically help to define topics. There is a general list of stop words such as: “a”; “a’s”; “able”; “about”; “above”... We also remove punctuation and capitalisation. The documents need to then be transformed into a document-term-matrix. This is essentially a table with a column of the number of times each term appears in each document.

After the dataset is ready, the R package ‘topicmodels’ by [Grün and Hornik \(2011\)](#) can be used to implement LDA and approximate the posterior. It does this using Gibbs sampling or the variational expectation-maximization algorithm. Following [Steyvers and Griffiths \(2006\)](#) and [Darling \(2011\)](#), the Gibbs sampling process attempts to find a topic for a particular term in a particular document, given the topics of all other terms for all other documents. Broadly, it does this by first assigning every term in every document to a random topic, specified by Dirichlet priors with  $\alpha = \frac{50}{K}$  and  $\eta = 0.1$  ([Steyvers and Griffiths \(2006\)](#) recommends  $\eta = 0.01$ ), where  $\alpha$  refers to the distribution over topics and  $\eta$  refers to the distribution over terms ([Grün and Hornik \(2011\)](#), p.7). It then selects a particular term in a particular document and assigns it to a new topic based on the conditional distribution where the topics for all other terms in all documents are taken as given ([Grün and Hornik \(2011\)](#), p.6):

$$p(z_{d,n} = k | w_{1:D,1:N}, z'_{d,n}) \propto \frac{\lambda'_{n \rightarrow k} + \eta}{\lambda'_{\cdot \rightarrow k} + V\eta} \frac{\lambda'^{(d)}_{n \rightarrow k} + \alpha}{\lambda'^{(d)}_{\cdot - i} + K\alpha}$$

where  $z'_{d,n}$  refers to all other topic assignments;  $\lambda'_{n \rightarrow k}$  is a count of how many other times that term has been assigned to topic  $k$ ;  $\lambda'_{\cdot \rightarrow k}$  is a count of how many other times that any term has been assigned to topic  $k$ ;  $\lambda'^{(d)}_{n \rightarrow k}$  is a count of how many other times that term has been assigned to topic  $k$  in that particular document; and  $\lambda'^{(d)}_{\cdot - i}$  is a count of how many other times that term has been assigned in that document. Once  $z_{d,n}$  has been estimated, then estimates for the distribution of words into topics and topics into documents can be backed out.

This conditional distribution assigns topics depending on how often a term has been assigned to that topic previously, and how common the topic is in that document (Steyvers and Griffiths (2006)). The initial random allocation of topics means that the results of early passes through the corpus of document are poor, but given enough time the algorithm converges to an appropriate estimate.

#### 18.3.4 Warnings and extensions

The choice of the number of topics,  $k$ , affects the results, and must be specified *a priori*. If there is a strong reason for a particular number, then this can be used. Otherwise, one way to choose an appropriate number is to use a test and training set process. Essentially, this means running the process on a variety of possible values for  $k$  and then picking an appropriate value that performs well.

One weakness of the LDA method is that it considers a ‘bag of words’ where the order of those words does not matter (Blei (2012)). It is possible to extend the model to reduce the impact of the bag-of-words assumption and add conditionality to word order. Additionally, alternatives to the Dirichlet distribution can be used to extend the model to allow for correlation. For instance, in Hansard topics related the army may be expected to be more commonly found with topics related to the navy, but less commonly with topics related to banking.

---

## 18.4 Word embedding

---

### 18.5 Conclusion

Using text as data is exciting because of the quantity and variety of text that is available to us. In general, dealing with text datasets is messy. There is a lot of cleaning and preparation that is typically required. Often text datasets are large. As such, having a workflow in place, in which you work in a reproducible way, simulating data first, and then clearly communicating your findings becomes critical, if only to keep everything organised in your own mind. Nonetheless, it is an exciting area, and I encourage you to regularly use text analysis where possible.

In terms of next steps there are two, related, concerns: data and analysis.

In terms of data there are many places to get large amounts of text data relatively easily, including:

- The r package `rtweets` makes it easy to get Twitter data (although typically this is going to be looking forward from when you start using it, rather than being able to look back). Plenty of people at U of T work with Twitter data including Jia Xue in the iSchool, and Ludovic Rheault in political science.
- The inside Airbnb dataset that we used earlier provides text from reviews.
- We've seen the `gutenbergr` package already in these notes, which provides easy access to text from Project Gutenberg.
- We've seen scraping of Wikipedia, but if you are going to do a bit of this then you may find it better to use a package, for instance `WikipediR`.

In terms of analysis:

- Start by going through the tidytext book, `tidytext`, as it has a lot of nice explanations, code, and examples.
  - It would then be worthwhile working through the `Quanteda` package `quanteda` tutorials.
  - Finally, consider packages such as `text2vec`, and `spacyr`.
- 

## 18.6 Exercises and tutorial

### 18.6.1 Exercises

### 18.6.2 Tutorial

---

---

## **Part VI**

# **Enrichment**

---

---



# 19

---

## *Using the cloud*

---

**STATUS:** Under construction.

### Required reading

- 

### Recommended reading

- Edmondson, Mark, 2020, ‘googleComputeEngineR documentation’, version 0.3.0.9000, freely available at: <https://cloudyr.github.io/googleComputeEngineR/>.
- McDermott, Grant R., 2020, ‘Cloud computing with Google Compute Engine’, *Data Science for Economists*, freely available at: <https://raw.githubusercontent.com/uo-ec607/lectures/master/14-gce/14-gce.html>.
- Morris, Mitzi, 2020, ‘Stan Notebooks in the Cloud’, freely available at: [https://mc-stan.org/users/documentation/case-studies/jupyter\\_colab\\_notebooks\\_2020.html](https://mc-stan.org/users/documentation/case-studies/jupyter_colab_notebooks_2020.html).

### Key concepts/skills/etc

- Benefits/costs of cloud.
  - Getting started in the cloud.
  - Starting virtual machines with R Studio.
  - Stopping virtual machines.
- 

### 19.1 Introduction

Cloud benefits: - Costs can be reduced, or more easily amortized. - Can scale as you need. - Many platforms are already sorted out e.g. R Studio just works.

I stole this from someone and I can't remember who, but the cloud is another name for ‘someone else’s computer’. That’s it. Nonetheless, learning to use someone else’s computer can be great for a number of reasons including:

- 1) Scalability: It can be quite expensive to buy a new computer, especially if you only need it to run something every now and then, but

by using someone else's computer, you can just rent for a few hours or days.

- 2) Portability: If you can shift your analysis workflow from your laptop to the cloud, then that suggests that you are likely doing good things in terms of reproducibility and portability. At the very least, your code is capable of running on your laptop and the cloud.
- 3) Set-and-forget: If you are doing something that will take a while, then it can be great to not have to worry about your laptop's fan running overnight, or your partner/baby/pet/housemate/etc accidentally closing your computer, or not being able to watch Netflix on that same computer.

When you use the cloud you are running your code on a 'virtual machine'. This is a part of a larger bunch of computers that has been designed to act like a computer with specific features. For instance you may specify that your virtual machine has 8 GB RAM, 128 storage, and 4 CPUs. Your VM would then act like a computer with those specifications. The cost to use cloud options increases based on the specifications of the virtual machine that you choose.

There are a few downsides:

- Cost: While most cloud options are cheap, they are rarely free. (While there are free options, they tend to not be very powerful, and so you end up having to pay to get a computer that is better than your laptop.) To give you an idea of cost, when I use AWS, I typically end up spending five to ten dollars for a couple of days. So it's fairly cheap, but it's not nothing. It's also pretty easy to accidentally forget about something and run up an unexpected bill, especially initially.
- Public: It is pretty easy to make mistakes and accidentally make everything public.
- Time: It takes time to get set-up and comfortable on the cloud.

In these notes we are going to introduce the cloud starting with some options that pretty much anyone can (and should) take advantage of: Google Colab; and then moving to more general cloud options including Google Compute Engine, AWS, and Azure, which may be useful to some of you in some cases. If you want to get a job in industry, then the advice of pretty much every speaker from industry at the Toronto Data Workshop is that you learn at least one of those cloud options. For instance, Munich Re is an Azure shop, Receptiviti uses AWS, etc.

## 19.2 Google Colab

Google Colab is similar to R Studio Cloud, in that it is set-up to allow you to just log in and get started. In this case, you need a Google account. It's better than R Studio because they have more resources to put into its development and you can use GPUs, but on the other hand it is designed for Python, and while we can use it for R, it's not really focused on that.

To get started you need to tell Google Colab that you want to use R. You can do this by using this: <https://colab.research.google.com/notebook#create=true&language=r>.

At this point you have a Jupyter notebook open that will run R. (But it is not a R Markdown document.) You can install packages as normal, e.g. `install.packages("tidyverse")`, and then call the package e.g. `library(tidyverse)`.

Google Colab is a good option if you have a good reason for using the broader capabilities that it has. If you want to go deeper into that then the Morris reading has a bunch of options that you can explore, but as Morris puts it ‘Colab is a gateway drug - for large-scale processing pipelines you’ll need to move up to Google Cloud Platform or one of its competitors AWS, Azure, etc.’ and that is what we will do now.

---

## 19.3 AWS

Amazon Web Services is a cloud service from Amazon. To get started you need an AWS Developer account which you can create here: <https://aws.amazon.com/developer/>.

After you have created an account, you need to select a region where the computer that you will access is located. After this, you will want to “Launch a virtual machine” (with EC2).

The first step is to choose an Amazon Machine Image (AMI). This provides the details of the computer that you will be using. For instance, your local computer may be a MacBook running Catalina. Helpfully, Louis Aslett provides a bunch of these already set up - [http://www.louisaslett.com/RStudio\\_AMI/](http://www.louisaslett.com/RStudio_AMI/). You can either select the code for the region that you registered for, or you can click on the link. The benefit of this AMI is that they are set-up specifically for R Studio, however the trade-off is that they are a little out-dated, as they were compiled in May 2019.

In the next step you can choose how powerful the computer will be. The free tier has a fairly basic computer, but you can choose better ones when you need them. At this point you can pretty much just launch the instance. If you start using AWS more seriously then you should look into different security settings.

Your instance is now running. You can go to it by pasting the ‘public DNS’ into a browser. The username is ‘rstudio’ and the password is your instance ID.

You should have R Studio running, which is exciting. The first thing to do is probably to change the default password using the instructions in the instance.

You don’t need to install, say, the tidyverse, instead you can just call the library and keep going. You can see the list of packages that are installed with `installed.packages()`. For instance, `rstan` is already installed. And you can use GPUs if you want.

Perhaps as important as being able to start an AWS instance is being able to stop it (so that you don’t get billed). The free tier is pretty great, but you do need to turn it off. To stop an instance, in the AWS instances page, select it, then ‘Actions -> Instance State -> Terminate’.

---

## 19.4 Google Compute Engine

The main R package related to Google Compute Engine seems to be: `google-ComputeEngineR`.

The reading from Grant McDermott is a pretty good walk-through.

---

## 19.5 Azure

There are a bunch of R packages related to Azure here: <https://github.com/Azure/AzureR>.

## **19.6 Exercises and tutorial**

### **19.6.1 Exercises**

### **19.6.2 Tutorial**



# 20

---

## *Deploying models*

---

**STATUS:** Under construction.

### Required reading

- Chip Huyen, 2020, ‘Machine learning is going real-time’, 27 December, <https://huyenchip.com/2020/12/27/real-time-machine-learning.html>.

### Required viewing

- Blair, James, 2019, ‘Democratizing R with Plumber APIs’, RStudio Conference, 24 January, <https://www.rstudio.com/resources/rstudioconf-2019/democratizing-r-with-plumber-apis/>.
- Nolis, Heather, and Jacqueline Nolis, ‘We’re hitting R a million times a day so we made a talk about it’, RStudio Conference, 30 January, <https://www.rstudio.com/resources/rstudioconf-2020/we-re-hitting-r-a-million-times-a-day-so-we-made-a-talk-about-it/>.

### Recommended reading

- 

### Key concepts/skills/etc

- Putting models into production requires a different set of skills to building a model. We need a familiarity with some cloud provider, APIs, and of course modelling. But the biggest difficulty, for me, is getting things set-up.

### Key libraries

- `plumber`
- `shiny`

### Key functions

- 

### Quiz

- 1.

---

## 20.1 Introduction

A key troupe against R is that it's not for production. I'm not here to convince you one way or another, however in this section we will go through a bunch of different tools that would allow you to do a lot in R if you wanted. The topics that we cover are:

- SQL databases.
- Docker.
- Plumber and APIs for models.
- Shiny
- Packages

The general idea here is that you need to know the whole workflow. To this point, you've been able to scrape some data from a website, bring some order to that chaos, make some charts, appropriately model it, and write this all up. In most academic settings that is more than enough. But in many industry settings we're going to want to use the model to do something. For instance, set-up a website that allows your model to be used to generate an insurance quote given several inputs.

One way to deploy your model is to use Shiny, and we have seen examples of this earlier in the notes. That enables an individual to use your model. But it doesn't really scale very well. For instance, if we wanted to sell our model forecasts to other businesses, then they might have their own way in which they would like users to interact with the results. The general problem is that we want our model results available to other machines and for that we will want to make an APIs.

---

## 20.2 Packages

To this point we've largely been using R Packages to do things for us. However, another way is to have them loaded

DoSS Toolkit

## 20.3 Shiny

---

## 20.4 Plumber and model APIs

### 20.4.1 Hello Toronto

The general idea behind the `plumber` package (Schloerke and Allen, 2021) is that we can train a model and make it available via an API that we can call when we want a forecast. It's pretty great.

Just to get something working, let's make a function that returns 'Hello Toronto' regardless of the output. Open a new R file, add the following, and then save it as 'plumber.R' (you may need to install the `plumber` package if you've not done that yet).

```
library(plumber)

@get /print_toronto
print_toronto <- function() {
 result <- "Hello Toronto"
 return(result)
}
```

After that is saved, in the top right of the editor you should get a button to 'Run API'. Click that, and your API should load. It'll be a 'Swagger' application, which provides a GUI around our API. Expand the GET method, and then click 'Try it out' and 'Execute'. In the response body, you should get 'Toronto'.

To more closely reflect the fact that this is an API designed for computers, you can copy/paste the 'request HTML' into a browser and it should return 'Hello Toronto'.

### 20.4.2 Local model

Now, we're going to update the API so that it serves a model output, given some input. We're going to follow [Buhr \(2017\)](#) fairly closely.

At this point, I'd recommend starting a new R Project. To get started, let's simulate some data and then train a model on it. In this case we're interested in forecasting how long a baby may sleep overnight, given we know how long they slept during their afternoon nap.

```

library(tidyverse)
set.seed(853)

number_of_observations <- 1000

baby_sleep <-
 tibble(afternoon_nap_length = rnorm(number_of_observations, 120, 5) %>% abs(),
 noise = rnorm(number_of_observations, 0, 120),
 night_sleep_length = afternoon_nap_length * 4 + noise,
)

baby_sleep %>%
 ggplot(aes(x = afternoon_nap_length, y = night_sleep_length)) +
 geom_point(alpha = 0.5) +
 labs(x = "Baby's afternoon nap length (minutes)",
 y = "Baby's overnight sleep length (minutes)") +
 theme_classic()

```

Let's now use `tidymodels` to quickly make a dodgy model.

```

set.seed(853)
library(tidymodels)

baby_sleep_split <- rsample::initial_split(baby_sleep, prop = 0.80)
baby_sleep_train <- rsample::training(baby_sleep_split)
baby_sleep_test <- rsample::testing(baby_sleep_split)

model <-
 parsnip::linear_reg() %>%
 parsnip::set_engine(engine = "lm") %>%
 parsnip::fit(night_sleep_length ~ afternoon_nap_length,
 data = baby_sleep_train
)

write_rds(x = model, file = "baby_sleep.rds")

```

At this point, we have a model. One difference from what you might be used to is that we've saved the model as an 'rds' file. We are going to read that in.

Now that we have our model we want to put that into a file that we will use the API to access, again called 'plumber.R'. And we also want a file that sets up the API, called 'server.R'. So make an R script called 'server.R' and add the following content:

```
library(plumber)

serve_model <- plumb("plumber.R")
serve_model$run(port = 8000)
```

Then in ‘plumber.R’ add the following content:

```
library(plumber)
library(tidyverse)

model <- readRDS("baby_sleep.rds")

version_number <- "0.0.1"

variables <-
list(
 afternoon_nap_length = "A value in minutes, likely between 0 and 240.",
 night_sleep_length = "A forecast, in minutes, likely between 0 and 1000."
)

@param afternoon_nap_length
@get /survival
predict_sleep <- function(afternoon_nap_length=0) {
 afternoon_nap_length = as.integer(afternoon_nap_length)

 payload <- data.frame(afternoon_nap_length=afternoon_nap_length)

 prediction <- predict(model, payload)

 result <- list(
 input = list(payload),
 response = list("estimated_night_sleep" = prediction),
 status = 200,
 model_version = version_number
)

 return(result)
}
```

Again, after you save the ‘plumber.R’ file you should have an option to ‘Run API’. Click that and you can try out the API locally in the same way as before.

### 20.4.3 Cloud model

To this point, we've got an API working on our own machine, but what we really want to do is to get it working on a computer such that the API can be accessed by anyone. To do this we are going to use DigitalOcean - <https://www.digitalocean.com>. It is a charged service, but when you create an account, it will come with \$100 in credit, which will be enough to get started.

This set-up process will be a pain and take some time, but you only need to do it once. Install two additional packages that will assist here:

- `plumberDeploy` (Allen, 2021).
- `analogsea` (Chamberlain et al., 2021).

```
install.packages("plumberDeploy")
remotes::install_github("sckott/analogsea")
```

Now we need to connect your local computer with your DigitalOcean account. Get started with:

```
analogsea::account()
```

Now you need to authenticate the connection and this is done using a SSH public key. You can do this using:

```
analogsea::key_create()
```

But if this is your first time doing this then it may be more useful to have a visual process, in which case follow the instructions here: <https://docs.digitalocean.com/products/droplets/how-to/add-ssh-keys/to-account/>. What you want is to have a 'pub' file on your computer. Then copy the public key aspect in that file, and add it to the SSH keys section in the account security settings. When you have the key on your local computer then you can check this using:

```
ssh::ssh_key_info()
```

Again, this will all take a while to validate. DigitalOcean calls every computer that you start a 'droplet'. So if you start three computers, then you'll have started three droplets. You can check the droplets that you have running using:

```
analogsea::droplets()
```

If everything is set-up properly, then this will print the information about all droplets that you have associated with your account (which at this point, is probably none).

To create a droplet, you run:

```
id <- plumberDeploy::do_provision(example = FALSE)
```

Then you'll get asked for your SSH passphrase and then it'll just set-up a bunch of things. After this we're going to need to install a whole bunch of things onto our droplet:

```
analogsea::install_r_package(droplet = id, c("plumber",
 "remotes",
 "here"))
analogsea::debian_apt_get_install(id, "libssl-dev",
 "libsodium-dev",
 "libcurl4-openssl-dev")
analogsea::debian_apt_get_install(id,
 "libxml2-dev")

analogsea::install_r_package(id, c("config",
 "httr",
 "urltools",
 "plumber"))

analogsea::install_r_package(id, c("xml2"))
analogsea::install_r_package(id, c("tidyverse"))

analogsea::install_r_package(id, c("tidymodels"))
```

And then when that is finally set-up (it'll seriously take 30 min or so) we can deploy our API!

```
plumberDeploy::do_deploy_api(droplet = id,
 path = "example",
 localPath = getwd(),
 port = 8000,
 docs = TRUE,
 overwrite=TRUE)
```

---

**20.5 Exercises and tutorial****20.5.1 Exercises****20.5.2 Tutorial**

# 21

---

## *Efficiency*

---

**STATUS:** Under construction.

**Required reading**

- 

**Required viewing**

- 

**Recommended reading**

- 

**Key concepts/skills/etc**

- 

**Key libraries**

- 

**Key functions**

- 

**Quiz**

1.

---

## 21.1 Introduction

---

## 21.2 Data efficiency

### 21.2.1 SQL

### 21.2.2 Feather

---

## 21.3 Code efficiency

By and large, worrying about performance is a waste of time. For the most part you are far better off, just pushing things into the cloud, letting them run for a reasonable time, and using that time to worry about other aspects of your pipeline. However, eventually this becomes unfeasible. For me, this is when something takes more than a day to run because it just becomes a pain. There is rarely a most common area for obvious performance gains. Instead you need to learn to measure and then cut.

Being fast is valuable but it's mostly about you being able to iterate fast not your code running fast. If you find that the speed at which your code completes is a bottle neck then shard. Then throw more machines at it. Then shard again. Then throw more machines at it.

---

## 21.4 Code refactoring

Some baby examples, focused on data science, along the lines of this:  
<https://indrajeetpatil.github.io/Refactoring-ggstatsplot/refactoring-ggstatsplot#1>

Start with an example of bad code, and then how it gets fixed.

### 21.4.1 Measure

Using `tic()` and `tic()`.

Measuring

---

## **21.5 Experimental efficiency**

Multi-armed bandit

---

---

## **21.6 Other languages**

**21.6.1 Python**

---

**21.6.2 Julia**

---

## **21.7 Exercises and tutorial**

**21.7.1 Exercises**

**21.7.2 Tutorial**



# 22

---

## *Concluding remarks, open issues, next steps*

---

**STATUS:** Under construction.

---

### 22.1 Concluding remarks

There's an old saying, something along the lines of 'may you live in interesting times'. I'm not sure if every generation feels this, but we sure live in interesting times. In this book, I have tried to convey some essentials that I think would allow you to contribute. But we are just getting started.

I'm 35 and so I am in the 'data science didn't exist when I was an undergraduate' generation. In a little over a decade data science has gone from something that barely existed to a defining part of academia and industry. What does that imply for you? It may imply that one should not just be making decisions that optimize for what data science looks like right now, but also what could happen. While that's a little difficult, that's also one of the things that makes data science so exciting. That might mean choices like:

- taking courses on fundamentals, not just fashionable applications;
- reading books, not just whatever is trending; and
- trying to be at the intersection of at least a few different areas, rather than hyper-specialized.

I'm just someone who likes to play with data using R. A decade ago I wouldn't have fit into any particular department. I'm lucky that these days there is space in data science for someone like me. And the nice thing about what we now call data science is that there's space for you as well.

Data science needs diversity. Data science needs your intelligence and enthusiasm. It needs you to be in the room, and able to make contributions. We live in interesting times and it's just such an exciting time to be enthusiastic about data. I can't wait to see what you build.

---

## 22.2 Some issues

1. How do we write unit tests for data science?

### **UPDATE to add in functional tests and stuff**

One thing that working with real computer scientists has taught me is the importance of unit tests. Basically this just means writing down the small checks that we do in our heads all the time. Like if we have a column that purports to the year, then it's unlikely that it's a character, and it's unlikely that it's an integer larger than 2500, and it's unlikely that it's a negative integer. We know all this, but writing unit tests has us write this all down.

In this case it's obvious what the unit test looks like. But more generally, we often have little idea what our results should look like if they're running well. The approach that I've taken is to add simulation—so we simulate reasonable results, write unit tests based on that, and then bring the real data to bear and adjust as necessary. But I really think that we need extensive work in this area because the current state-of-the-art is lacking.

2. What happened to the machine learning revolution?

I don't understand what happened to the promised machine learning revolution in social sciences. Specifically, I'm yet to see any convincing application of machine learning methods that are designed for prediction to a social sciences problem where what we care about is understanding. I would like to either see evidence of them or a definitive thesis about why this can't happen. The current situation is untenable where folks, especially those in fields that have been historically female, are made to feel inferior even though their results are no worse.

3. How do we think about power?

As someone who learnt statistics from economists, but now is partly in a statistics department, I do think that everyone should learn statistics from statisticians. This isn't anything against economists, but the conversations that I have in the statistics department about what statistical methods are and how they should be used are very different to those that I've had in other departments.

I think the problem is that people outside statistics, treat statistics as a recipe in which they follow various steps and then out comes a cake. With regard to 'power'—it turns out that there were a bunch of instructions that no one bothered to check—they turned the oven on to some temperature without

checking that it was 180C, and that's fine because whatever mess came out was accepted because the people evaluating the cake didn't know that they needed to check the temperature had been appropriately set. (I'm ditching this analogy right now).

As you know, the issue with power is related to the broader discussion about p-values, which basically no one is taught properly, because it would require changing an awful lot about how we teach statistics i.e. moving away from the recipe approach.

And so, my specific issue is that people think that statistics is a recipe to be followed. They think that because that's how they are trained especially in social sciences like political science and economics, and that's what is rewarded. But that's not what these methods are. Instead, statistics is a collection of different instruments that let us look at our data in a certain way. I think that we need a revolution here, not a metaphorical tucking in of one's shirt.

---

### 22.3 Next steps

This book has covered much ground, and while we are toward the end of it, as the butler Stevens is told in the novel *The Remains of the Day* (Ishiguro, 1989):

---

The evening's the best part of the day. You've done your day's work. Now you can put your feet up and enjoy it.

---

Chances are there are aspects that you want to explore further, building on the foundation that you have established. If so, then I've accomplished what I set out to do.

If you were new to data science at the start of this book, then the next step would be to backfill that which I skipped over, and I would recommend Timbers et al. (2022). After that, you should learn more about R in terms of data science by going through Wickham and Grolemund (2017). To deepen your understanding of R itself, go next to Wickham (2019a).

If you're interested in learning more about causality then start with Cunningham (2021) and Huntington-Klein (2021).

If you're interested to learn more about statistics then begin with McElreath

(2020), and then backfill with [Johnson et al. \(2022\)](#) and solidify the foundation with [Gelman et al. \(2014\)](#). You should probably also backfill some of the fundamentals around probability, starting with [Wasserman \(2005\)](#).

There is only one next natural step if you're interested in learning more about statistical (what's come to be called machine) learning and that's [James et al. \(2017\)](#) followed by [Friedman et al. \(2009\)](#).

If you're interested in sampling then the next book to turn to is [Lohr \(2019\)](#). To deepen your understanding of surveys and experiments, go next to [Gerber and Green \(2012\)](#) in combination with [Kohavi et al. \(2020\)](#).

For graphs turn to [Healy \(2018\)](#).

Writing go to...

Thinking through production and SQL and things like, a next natural step is...

We often hear the phrase let the data speak. Hopefully by this point you understand that never happens. All that we can do is to acknowledge that we are the ones using data to tell stories, and strive and seek to make them worthy.

# A

---

*Oh you think and shoulders and datasets*

---

*This is just a holding place for this content while it is being developed.*

---

---

## A.1 Oh, you think we have good data on that!

---

Oh, you think we have good data on that! Migration.

---

---

Oh, you think we have good data on that! Weather stations

---

---

Oh, you think we have good data on that! Olympics events.  
Who decides on the scoring?. Who does the timing?

---

---

Oh, you think we have good data on that! Personality  
scores. Myers Briggs and Big 5 more generally.

---

---

---

---

**Oh, you think we have good data on that!** Cause of death

---

---

**Oh, you think we have good data on that!** City boundaries.  
What constitutes ‘Atlanta’? Different definitions - metro, X, Y.  
(also an issue in countries with boundaries changing over time)

---

---

**Oh, you think we have good data on that!** Timing

---

---

## A.2 Shoulders of giants

Chapter 2:

---

---

**Shoulders of giants** Robert Gentleman and Ross Ihaka

---

---

Chapter 5:

---

---

**Shoulders of giants** Xiao-Li Meng

---

---

- Andrew Gelman

- Barbara Bailar
  - Daniela Witten
  - Elizabeth Scott
  - Evelyn Kitagawa
  - Gertrude Mary Cox
  - Hadley Wickham
  - John Tukey
  - Katherine Wallman
  - Nancy Reid
  - Simon Kuznets
  - Stella Cunliffe
  - Rob Tibshirani
  - Timnit Gebru
- 

### A.3 Possible datasets

- <https://som.yale.edu/faculty-research/our-centers/international-center-finance/data>
- Alex cookson
- David Andrew's book
- Tidy census
- <https://stat.ethz.ch/R-manual/R-patched/library/datasets/html/00Index.html>
- <https://www.historicalstatistics.org/>
- <https://data.cityofberkeley.info/browse?limitTo=datasets&utf8>
- <https://data.gov.hk/en-datasets/category/education>
- <https://data.rijksmuseum.nl/object-metadata/download/>
- World bank <https://data.worldbank.org/> eg development indicators
- South sea bubble
- OECD
- Aer R package and that paper?
- CESr
- Paspaley
- Canlang
- Fred - does that have an api?
- 538
- The Economist
- <https://pds.nasa.gov/datasearch/subscription-service/SS-Release.shtml>
- <https://github.com/BuzzFeedNews/nics-firarm-background-checks>
- The markup
- Tom Cardoso

- List of APIs: [https://bookdown.org/paul/apis\\_for\\_social\\_scientists/](https://bookdown.org/paul/apis_for_social_scientists/)

# B

---

## Papers

---

---

### B.1 Paper 1

#### B.1.1 Task

Working individually and in an entirely reproducible way, please find a dataset of interest on Open Data Toronto – <https://open.toronto.ca> – and write a short paper telling a story about the data.

#### B.1.2 Guidance

- Find a dataset of interest on Open Data Toronto<sup>1</sup> and download it in a reproducible way using `opendatatoronto` ([Gelfand, 2020](#)).
- Create a folder with appropriate sub-folders, add it to GitHub, and then prepare a PDF using R Markdown with these sections (you are welcome to use this starter folder: [https://github.com/RohanAlexander/starter\\_folder](https://github.com/RohanAlexander/starter_folder)):
  - title,
  - author,
  - date,
  - abstract,
  - introduction,
  - data, and
  - references.
- In the data section thoroughly and precisely discuss the source of the data and the bias this brings (ethical, statistical, and otherwise). Comprehensively describe and summarize the data using text and at least one graph and one table. Graphs must be made in `ggplot2` ([Wickham, 2016](#)) and tables must be made using `knitr` ([Xie, 2021](#)) (with or without `kableExtra` ([Zhu, 2020](#))). Graphs must show the actual data, or as close to it as possible, not summary statistics. Make sure to cross-reference graphs and tables.
- Add references by using a bib file. Be sure to reference R and any R packages you use, as well as the dataset. Check that you have referenced everything. Strong submissions will draw on related literature and would also reference

---

<sup>1</sup><https://open.toronto.ca>

those. There are various options in R Markdown for references style; just pick one that you are used to.

- Go back and write an introduction. This should be two or three paragraphs. The last paragraph should set out the remainder of the paper.
- Add an abstract. This should be three or four sentences. If your abstract is longer than four sentences, then you need to think a lot about whether it is too long. It may be fine (there are always exceptions) but you should probably have a good reason. Your abstract must tell the reader your top-level finding. What is the one thing that we learn about the world because of your paper?
- Then add a descriptive title. ‘Paper 1’ is not descriptive and there should not be any sign this is a school paper.
- Add a link to your GitHub repo using a footnote.
- Check that your GitHub repo is well-organized, and add an informative README. Comment your code. Make sure that you have got at least one R script in there, in addition, to your R Markdown file.
- Pull this all together as a PDF and check that the paper is well-written and able to be understood by the average reader of, say, FiveThirtyEight. This means that you are allowed to use mathematical notation, but you must explain all of it in plain language. All statistical concepts and terminology must be explained. Your reader is someone with a university education, but not necessarily someone who understands what a p-value is.
- Check there is no evidence that this is a class assignment.
- Via Quercus, submit the PDF.

### B.1.3 Checks

- Check you have not included any R code or raw R output in the final PDF.
- Check that although you will probably have most of your code in the R Markdown, make sure that you have at least one R script in the ‘scripts’ folder.
- Check there is thoroughly commented code that directly creates your PDF. Do not ‘knit to html’ and then save as a PDF. Do not ‘knit to Word’ and then save as a PDF
- Check that your graphs, tables, and text are extremely clear, and of comparable quality to those of FiveThirtyEight.
- Check that the date is updated.
- Check your entire workflow is entirely reproducible.
- Check for typos.

### B.1.4 FAQ

- Can I use a dataset from Kaggle instead? No, because they have done the hard work for you.
- I cannot use code to download my dataset, can I just manually download it?

No, because your entire workflow needs to be reproducible. Please fix the download problem or pick a different dataset.

- How much should I write? Most students submit something in the two-to-six-page range, but it's really up to you. Be precise and thorough.
- My data is about apartment blocks/NBA/League of Legends so there's no ethical or bias aspect, what do I do? Please re-read the readings to better understand bias and ethics. If you really cannot think of something, then it might be worth picking a different dataset.
- Can I use Python? No. If you already know Python then it doesn't hurt to learn another language.
- Why do I need to cite R, when I don't need to cite Word? R is a free statistical programming language with academic origins so it's appropriate to acknowledge the work of others. It's also important for reproducibility.

### B.1.5 Rubric

- Go/no-go #1: R is cited - [1 'Yes', 0 'No']
  - Both referred to in the main content and included in the reference list.
  - If not, no need to continue marking, just give paper 0 overall.
- Title - [2 'Exceptional', 1 'Yes', 0 'Poor or not done']
  - An informative title is included.
  - Tell the reader what your story is, don't waste their time.
  - Ideally tell them what happens at the end of the story.
  - 'Problem Set X' is not an informative title. There should be no evidence this is a school paper.
- Author, date, and repo - [2 'Yes', 0 'Poor or not done']
  - The author, date of submission, and a link to a GitHub repo are clearly included. (The later likely, but not necessarily, through a statement such as: 'Code and data supporting this analysis is available at: [LINK](#)').
- Abstract - [4 'Exceptional', 3 'Great', 2 'Fine', 1 'Gets job done', 0 'Poor or not done']
  - An abstract is included and appropriately pitched to a general audience.
  - The abstract answers: 1) what was done, 2) what was found, and 3) why this matters (all at a high level).
- Introduction - [4 'Exceptional', 3 'Great', 2 'Fine', 1 'Gets job done', 0 'Poor or not done']
  - The introduction is self-contained and tells a reader everything they need to know, including putting it into a broader context.
  - Your introduction should provide a bit of broader context to motivate the reader, as well as providing a bit more detail about what you're interested in, what you did, what you found, why it's important, etc.
  - A reader should be able to read only your introduction and have a good idea about the research that you carried out and what you found.
  - It would be rare that you would have tables or figures in your introduc-

tion (again there are always exceptions but think deeply about whether yours is one).

- It must outline the structure of the paper.
- For instance (and this is just a rough guide) an introduction for a 10 page paper, should probably be about 3 or 4 paragraphs, or 10 per cent, but it depends on specifics.
- Data - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - When you discuss the dataset (in the data section) you should make sure to discuss at least:
    - 1) The source of the data.
    - 2) The methodology and approach that is used to collect and process the data.
    - 3) The population, the frame, and the sample (as appropriate).
    - 4) Information about how respondents were found. What happened to non-response?
    - 5) What are the key features, strengths, and weaknesses about the source generally.
  - You should thoroughly discuss the variables in the dataset that you use. Are there any that are very similar that you nonetheless don’t use? Did you construct any variables by combining various ones?
  - What do the data look like?
  - Plot the actual data that you’re using (or as close as you can get to it).
  - Discuss these plots and the other features of these data.
    - \* These are just some of the issues strong submissions will consider. Show off your knowledge. If this becomes too detailed, then you should push some of this to footnotes or an appendix.
    - \* ‘Exceptional’ means that when I read your submission I learn something about the dataset that I don’t learn from any other submission (within a reasonable measure of course).
- Numbering - [2 ‘Yes’, 0 ‘Poor or not done’]
  - All figures, tables, equations, etc are numbered and referred to in the text.
- Proofreading - [2 ‘Yes’, 0 ‘Poor or not done’]
  - All aspects of submission are free of noticeable typos.
- Graphs/tables/etc - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - You must include graphs and tables in your paper and they must be to a high standard.
  - They must be well formatted and camera-ready. They should be clear and digestible.
  - They must: 1) serve a clear purpose; 2) be fully self-contained through appropriate use of labels/explanations, etc; and 3) appropriately sized and colored (or appropriate significant figures in the case of stats).

- References - [4 ‘Perfect’, 3 ‘One minor issue’, 0 ‘Poor or not done’]
  - All data/software/literature/etc are appropriately noted and cited.
  - You must cite the software and software packages that you use.
  - You must cite the datasets that you use.
  - You must cite literature that you refer to (and you should refer to literature).
  - If you take a small chunk of code from Stack Overflow then add the page in a comment next to the code.
  - If you take a large chunk of code then cite it fully.
  - 3 means one minor issue. More than one minor issue receives 0.
- Reproducibility - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - The paper and analysis must be fully reproducible.
  - A detailed README is included.
  - All code should be thoroughly documented.
  - An R project is used. Do not use `setwd()`.
  - The code must appropriately read data, prepare it, create plots, conduct analysis, and generate documents. Seeds are used where needed.
  - Code must have a preamble etc.
  - You must appropriately document your scripts such that someone coming in could follow them.
  - Your repo must be thoroughly organized.
- General excellence - [3 ‘Exceptional’, 2 ‘Wow’, 1 ‘Huh, that’s interesting’, 0 ‘None’]
  - There are always students that excel in a way that is not anticipated in the rubric. This item accounts for that.

### B.1.6 Previous examples

Some examples of papers that well in the past include those by: Amy Farrow<sup>2</sup>, Morgaine Westin<sup>3</sup>, and Rachel Lam<sup>4</sup>.

---

<sup>2</sup>[inputs/pdfs/Mandatory\\_minimums-Amy\\_Farrow.pdf](#)

<sup>3</sup>[inputs/pdfs/Mandatory\\_minimums-Morgaine\\_Westin.pdf](#)

<sup>4</sup>[inputs/pdfs/Mandatory\\_minimums-Rachel\\_Lam.pdf](#)

## B.2 Paper 2

### B.2.1 Task

[ADD STUFF aBOUT REPLICATION LAB]

- Working as part of a small team of 1-3 people, and in an entirely reproducible way, please pick a paper to reproduce from an approved list and then write a short paper telling a story based on this. Your story should both talk about the (reproduced) findings, but also (a bit more ‘meta’) about what you learnt from the process.

### B.2.2 Guidance

- Working as part of a team of 1-3 people, prepare a PDF in R Markdown with the following features:
  - title,
  - author/s,
  - date,
  - abstract,
  - introduction,
  - data,
  - model,
  - results,
  - discussion, and
  - references.
- In the discussion section and any other relevant section, please be sure to discuss ethics and bias with reference to relevant literature.
- You should reproduce one of the following papers:
  - Barari, Soubhik, Christopher Lucas, and Kevin Munger, 2021, ‘Political Deepfake Videos Misinform the Public, But No More than Other Fake Media’, 13 January, <https://osf.io/cdfh3/>.
  - Cohn, Alain, Michel André Maréchal, David Tannenbaum, and Christian Lukas Zünd, 2019, ‘Civic Honesty Around the Globe’.
  - Liran Einav, Amy Finkelstein, Tamar Oostrom, Abigail Ostriker, Heidi Williams, 2020, ‘Screening and Selection: The Case of Mammograms’, *American Economic Review*.
  - Pons, Vincent, 2018, ‘Will a Five-Minute Discussion Change Your Mind? A Countrywide Experiment on Voter Choice in France’ *American Economic Review*.
  - Abramitzky, Ran, Leah Boustan, Katherine Eriksson, and Stephanie Hao. “Discrimination and the Returns to Cultural Assimilation in the Age of Mass Migration.” AEA Papers and Proceedings 110 (May 2020): 340–46. <https://doi.org/10.1257/pandp.20201090>.

- Chen, Yan, Ming Jiang, and Erin L. Krupka. “Hunger and the Gender Gap.” *Experimental Economics* 22, no. 4 (December 2019): 885–917. <https://doi.org/10.1007/s10683-018-9589-9>.
- Lise, Jeremy, and Fabien Postel-Vinay. “Multidimensional Skills, Sorting, and Human Capital Accumulation.” *American Economic Review* 110, no. 8 (August 2020): 2328–76. <https://doi.org/10.1257/aer.20162002>.
- Kolev, Julian, Yuly Fuentes-Medel, and Fiona Murray. “Gender Differences in Scientific Communication and Their Impact on Grant Funding Decisions.” *AEA Papers and Proceedings* 110 (May 2020): 245–49. <https://doi.org/10.1257/pandp.20201043>.
- Cowgill, Bo, Fabrizio Dell’Acqua, and Sandra Matz. “The Managerial Effects of Algorithmic Fairness Activism.” *AEA Papers and Proceedings* 110 (May 2020): 85–90. <https://doi.org/10.1257/pandp.20201035>.
- You should follow the lead of the author/s of the paper you’re reproducing, but thoroughly think about, and discuss, what is being done. Regardless of the particular model that you are using, and the (possibly lack of) extent to which this is done in the paper, your model must be well explained, thoroughly justified, explained as appropriate to the task at hand, and the results must be beautifully described.
- You must include a DAG (probably in the model section).
- You must have a discussion of power and experimental design (probably in the data section)
- Your paper must be well-written, draw on relevant literature, and show your statistical skills by explaining all statistical concepts that you draw on.
- You are welcome to use appendices for supporting, but not critical, material. Your discussion must include sub-sections that focus on three or four interesting points, and also sub-sections on weaknesses and next steps.
- In your report you must provide a link to a GitHub repo that fully contains your analysis. Your code must be entirely reproducible, documented, and readable. Your repo must be well-organised and appropriately use folders.
- Your graphs and tables must be of an incredibly high standard. Graphs and tables should be well formatted and report-ready. They should be clean and digestible. Furthermore, you should label and describe each table/figure.
- When you discuss the dataset (in the data section) you should make sure to discuss (at least):
  - Its key features, strengths, and weaknesses generally.
  - A discussion of the questionnaire - what is good and bad about it?
  - A discussion of the methodology including how they find people to take the survey; what their population, frame, and sample were; what sampling approach they took and what some of the trade-offs may be; what they do about non-response; the cost.
  - A discussion of the intervention and experimental design.
  - These are just some of the issues strong submissions will consider. Show

off your knowledge. If this becomes too detailed then you should push some of this to footnotes or an appendix.

- When you discuss your model (in the model section), you must be extremely careful to spell out the statistical model that you are using, defining and explaining each aspect and why it is important. (For a Bayesian model, a discussion of priors and regularization is almost always important.) You should mention the software that you used to run the model. You should be clear about model convergence, model checks, and diagnostic issues. How do the sampling and survey aspects that you discussed assert themselves in the modelling decisions that you make? Again, if it becomes too detailed then push some of the details to footnotes or an appendix. You have the original paper to guide you, but you will likely need to go well-beyond what is included.
- You should present model results, graphs, figures, etc, in the results section. This section should strictly relay results. Interpretation of these results and conclusions drawn from the results should be left for the discussion section.
- Your discussion should focus on your model results. Interpret them and explain what they mean. Put them in context. What do we learn about the world having understood your model and its results? What caveats could apply? To what extent does your model represent the small world and the large world (to use the language of McElreath, Ch 2)? What are some weaknesses and opportunities for future work? Additionally, as this is a reproduction you should include a sub-section on differences you found and difficulties that you had.
- Check that you have referenced everything. Strong submissions will draw on related literature in the discussion (and other sections) and would be sure to also reference those. The style of references does not matter, provided it is consistent.
- As a team, via Quercus, submit a PDF of your paper. Again, in your paper you must have a link to the associated GitHub repo. And you must include the R Markdown file that produced the PDF in that repo. And you must include the R Markdown file that produced the PDF in that repo. The repo must be well-organized and have a detailed README.
- A good way to work as a team would be to split up the work, so that one person is doing each section. The people doing the sections that rely on data (such as the analysis and the graphs) could just simulate it while they are waiting for the person putting together the data to finish.
- It is expected that your submission be well written and able to be understood by the average reader of say 538. This means that you are allowed to use mathematical notation, but you must be able to explain it all in plain English. Similarly, you can (and hint: you should) use survey, sampling, observational, and statistical terminology, but again you need to explain it. Your work should have flow and should be easy to follow and understand. To communicate well, anyone at the university level should be able to read

your report once and relay back the methodology, overall results, findings, weaknesses and next steps without confusion.

- Everyone in the team receives the same mark.
- There should be no evidence that this is a class assignment.

### B.2.3 Checks

- We have not just copy-pasted the code from the original paper, but instead have used that as a foundation to work from?

### B.2.4 FAQ

- Do I have to stay in the same group as the second paper? No. You're welcome to change. However, it's important that you don't change the second paper group on Quercus - be sure to only change the third paper group.
- Can we switch groups for the third paper? Yes.
- How much should I write? Most students submit something in the 10-to-15-page range, but it's really up to you. Be precise and thorough.
- My paper doesn't have a DAG, what do I do? You need to make the DAG.

### B.2.5 Rubric

- Go/no-go #1: R is cited - [1 'Yes', 0 'No']
  - Both referred to in the main content and included in the reference list.
  - If not, no need to continue marking, just give paper 0 overall.
- Title - [2 'Exceptional', 1 'Yes', 0 'Poor or not done']
  - An informative title is included.
  - Tell the reader what your story is - don't waste their time.
  - Ideally tell them what happens at the end of the story.
  - 'Problem Set X' is not an informative title. There should be no evidence this is a school paper.
- Author, date, and repo - [2 'Yes', 0 'Poor or not done']
  - The author, date of submission, and a link to a GitHub repo are clearly included. (The later likely, but not necessarily, through a statement such as: 'Code and data supporting this analysis is available at: [LINK](#)').
- Abstract - [4 'Exceptional', 3 'Great', 2 'Fine', 1 'Gets job done', 0 'Poor or not done']
  - An abstract is included and appropriately pitched to a general audience.
  - The abstract answers: 1) what was done, 2) what was found, and 3) why this matters (all at a high level).
  - If your abstract is longer than four sentences then you need to think a lot about whether it is too long. It may be fine (there are always exceptions) but you should probably have a good reason.
  - Your abstract must tell the reader your top-level finding. What is the one thing that we learn about the world because of your paper?

- Introduction - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - The introduction is self-contained and tells a reader everything they need to know, including putting it into a broader context.
  - Your introduction should provide a bit of broader context to motivate the reader, as well as providing a bit more detail about what you’re interested in, what you did, what you found, why it’s important, etc.
  - A reader should be able to read only your introduction and have a good idea about the research that you carried out.
  - It would be rare that you would have tables or figures in your introduction (again there are always exceptions but think deeply about whether yours is one).
  - It must outline the structure of the paper.
  - For instance (and this is just a rough guide) an introduction for a 10 page paper, should probably be about 3 or 4 paragraphs, or 10 per cent, but it depends on specifics.
- Data - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - You should thoroughly discuss the variables in the dataset that you use. Are there any that are very similar that you nonetheless don’t use? Did you construct any variables by combining various ones?
  - What do the data look like?
  - Plot the actual data that you’re using (or as close as you can get to it).
  - Discuss these plots and the other features of these data.
  - These are just some of the issues strong submissions will consider. Show off your knowledge. If this becomes too detailed, then you should push some of this to footnotes or an appendix.
  - ‘Exceptional’ means that when I read your submission I learn something about the dataset that I don’t learn from any other submission (within a reasonable measure of course).
- Model - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - The model is nicely written out, well-explained, justified, and appropriate.
  - When you discuss your model you must be extremely careful to spell out the statistical model that you are using defining and explaining each aspect and why it is important. Failure to do this suggests you don’t understand the model.
  - The model is appropriately complex - that is, not too simple, but not unnecessarily complicated.
  - The model has well-defined variables and these correspond to what is discussed in the data section.
  - The model needs to be written out in appropriate mathematical notation but also in plain English.

- Every aspect of that notation must be defined otherwise the most this section can receive is poor.
- The model makes sense based on the substantive area, and also the form of the model.
- If the model is Bayesian, then priors need to be defined and sensible.
- Discussion needs to occur around how features enter the model and why. For instance, (and these are just examples) why use ages rather than age-groups, why does province have a levels effect, why is gender categorical, etc?
- In general, in order to be adequate, there needs to be a clear justification that this is the model for the situation.
- The assumptions underpinning the model are clearly discussed.
- Alternative models, or variants, must be discussed and strengths and weaknesses made clear. Why was this model chosen?
- You should mention the software that you used to run the model.
- There is some evidence of thought about the circumstances in which the model may not be appropriate.
- There is evidence of model validation and checking, whether that is out of sample or comparison to a straw man or RMSE, test/training, or appropriate sensitivity checks.
- You should be clear about model convergence, model checks, and diagnostic issues, but if this becomes too detailed then you could push some of this to an appendix.
- Great answers would discuss things such as, how do the aspects that you discussed in the data section assert themselves in the modelling decisions that you make. Again if it becomes too detailed then push some of the details to footnotes or an appendix.
- Again, explain what your model is and what is going on with it.
- Results - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - Results will likely require summary statistics, tables, graphs, images, and possibly statistical analysis or maps.
  - To be clear, you should also have text associated with all these aspects.
  - Show the reader the results by plotting them. Talk about them. Explain them. That said, this section should strictly relay results.
- Discussion - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - Some questions that a good discussion would cover include (each of these would be a sub-section of something like half a page to a page):
    - \* What is done in this this paper?
    - \* What is something that we learn about the world?
    - \* What is another thing that we learn about the world?
    - \* What are some weaknesses of what was done?
    - \* What is left to learn or how should we proceed in the future?
- Numbering - [2 ‘Yes’, 0 ‘Poor or not done’]

- All figures, tables, equations, etc are numbered and referred to in the text.
- Proofreading - [2 ‘Yes’, 0 ‘Poor or not done’]
  - All aspects of submission are free of noticeable typos.
- Graphs/tables/etc - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - You must include graphs and tables in your paper and they must be to a high standard.
  - They must be well formatted and camera-ready They should be clear and digestible.
  - They must: 1) serve a clear purpose; 2) be fully self-contained through appropriate use of labels/explanations, etc; and 3) appropriately sized and coloured (or appropriate significant figures in the case of stats).
- References - [4 ‘Perfect’, 3 ‘One minor issue’, 0 ‘Poor or not done’]
  - All data/software/literature/etc are appropriately noted and cited.
  - You must cite the software and software packages that you use.
  - You must cite the datasets that you use.
  - You must cite literature that you refer to (and you should refer to literature).
  - If you take a small chunk of code from Stack Overflow then add the page in a comment next to the code.
  - If you take a large chunk of code then cite it fully.
  - 3 means one minor issue. More than one minor issue receives 0.
- Reproducibility - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - The paper and analysis must be fully reproducible.
  - A detailed README is included.
  - All code should be thoroughly documented.
  - An R project is used. Do not use `setwd()`.
  - The code must appropriately read data, prepare it, create plots, conduct analysis, generate documents, etc. Seeds are used where needed.
  - Code must have a preamble etc.
  - You must appropriately document your scripts such that someone coming in could follow them.
  - Your repo must be thoroughly organized and not contain extraneous files.
- General excellence - [3 ‘Exceptional’, 2 ‘Wow’, 1 ‘Huh, that’s interesting’, 0 ‘None’]
  - There are always students that excel in a way that is not anticipated in the rubric. This item accounts for that.

### B.3 ‘These numbers mean dial it up’

#### B.3.1 Task

Please consider this scenario:

---

You are employed as a junior data scientist at Petit Poll - a Canadian polling company. Petit Poll has a contract with a ‘client’ - an Ontario government department - to provide them with advice. In particular, the client wants to understand the effect of COVID shut-downs on restaurant businesses and has asked Petit Poll to design an experiment about some aspect of this.

---

Working as part of a small team of 1-3 people, and in an entirely reproducible way, please decide on an intervention, and some measurement strategies, and then write a short paper telling a story about the effect.

#### B.3.2 Guidance

- Working as part of a team of 1-3 people, prepare a PDF in R Markdown with the following features (you are welcome to use this starter folder: [https://github.com/RohanAlexander/starter\\_folder](https://github.com/RohanAlexander/starter_folder)):
  - title,
  - author/s,
  - date,
  - abstract,
  - introduction,
  - data,
  - discussion,
  - appendix with survey details, and
  - references.
- Decide on an intervention. Some aspects to address include:
  - How will it be designed and implemented?
  - What will be random about it?
  - How will you ensure the separation of treatment and non-treatment?
  - How long will it run?
- you will need to run surveys to gather information about your intervention. Decide on a survey methodology. Some aspects to address include:

- What is the population, frame, and sample?
- What sampling methods will you use and why? What are some of the statistical properties that the method brings to the table?
- How are you going to reach your desired respondents?
- How much do you estimate this will cost?
- What steps will you take to deal with non-response and how will non-response affect your survey?
- How are you going to protect respondent privacy?
- Remember to consider all of this in the context of your ‘client’ - for instance, what are they interested in?
- Develop a survey on a platform that was introduced in class, or another that you’re familiar with.
  - Be sure to test it yourselves. You will want to test this as much as possible, maybe even swap informally with another group?
- Now release the surveys into the (simulated) ‘field’.
  - Please do this by simulating an appropriate number of responses to your survey in R.
  - Don’t forget to simulate in relation to the intervention that you proposed.
  - Do you need two, or even more, surveys (and hence multiple sets of simulated results)?
- Show the results and discuss your ‘findings’. Everything must be entirely reproducible.
- You may wish to scrape some data and/or use open data sources to appropriately parameterize your simulations. Don’t forget to cite them when you do this.
- Use R Markdown to write a PDF report about all of this. Discuss your intervention, results and findings, your survey design and motivations, etc - all of it. You are writing a report that will eventually go to the client, so you must set the scene, and use language that demonstrates your command of statistical concepts but brings the reader along with you. Be sure to include graphs and tables and reference them in your discussion. Be sure to be clear about weaknesses and biases, and opportunities for future work.
- Your report must be well written. You are allowed to, and should, use mathematical notation and statistical concepts, but you must explain all of it in plain language. Similarly, you can, and should, use experimental/survey/sampling/observational data terminology, but again, you need to explain it.
- In the data section you should specify the intervention and data gathering methodology. You should also show your data, with tables and graphs as necessary. In addition to summaries, be sure to plot your raw data to the extent possible.
- Your graphs and tables must be of an incredibly high standard. Graphs and tables should be well formatted and report-ready. They should be

clean and digestible. Furthermore, you should label and describe each table/figure/equation.

- In the discussion section and any other relevant section, please be sure to discuss ethics and bias with reference to relevant literature. For instance, think about who is and who is not in your dataset. What are the statistical and ethical implications of this?
- Often folks struggle with the Discussion section of reports. For a 10 page report, we’re looking for about 2-3 pages of content. Here’s an example of sub-sections that you could include:
  - A sub-section containing a brief overview of the paper, and also how it fits into the literature and improves existing contributions.
  - Three sub-sections that detail the three main points that we learn about the world from the paper.
  - A sub-section on limitations, but discussed in a sophisticated way. So this means not just stating them, but explaining and justifying.
  - A sub-section on how it could be extended and future directions.
- Your client has stats graduates working for it who need to be impressed by the main content of the report, but also has people who barely know what an average is and these people need to be impressed also.
- Check that you have referenced everything, including R, R packages, and datasets. Strong submissions will draw on related literature and would be sure to also reference those. The style of references does not matter - there are various options in R Markdown - provided it is consistent and that `bibtex` has been used.
- Via Quercus, submit your PDF report. You must provide a link to the GitHub repo where the code that you used for this assignment lives. Comment. Your. Code. Your entire workflow must be entirely reproducible. Your repo should be clearly organised, and a useful README included. You must include a separate R script that accomplishes something (probably the simulations makes sense). And you must include the R Markdown file that produced the PDF in that repo.
- Please be sure to include a link to your survey/s in your report and screenshots of the survey/s in the appendix of your report.
- Everyone in the team receives the same mark.
- There should be no evidence that this is a class assignment.

### B.3.3 Checks

- Check you have not included any R code or raw R output in the final PDF.
- Check you have cited R and any R packages used.
- Check that although you will probably have most of your code in the R Markdown, make sure that you have at least one R script in the `scripts` folder.
- Check there is thoroughly commented code that directly creates your PDF.

Do not knit to html and then save as a PDF. Do not knit to Word and then save as a PDF

- Check that your graph and discussion are extremely clear, and of comparable quality to those of FiveThirtyEight.
- Check that the date is updated to the date of submission.
- Check your entire workflow is entirely reproducible.
- Check for typos.
- Check that you have got an appendix that details the survey/s and a link to the live survey.

#### **B.3.4 FAQ**

- Can I work by myself? Yes. But I recommend forming a group and the workload for the course assumes you will work on the second and third paper as part of a group of four.
- Can we switch groups for the third paper? Yes.
- How can I find a group? I will randomly create groups of four in Quercus. You are welcome to shift out of those groups and form your own groups if you'd like.
- Can I get a different mark to the rest of my group? No. Everyone in the group gets the same mark.
- I wrote my paper by myself, so can I be graded on a different scale? No. All papers are graded in the same way.
- How much should I write? Most students submit something in the 10-to-15-page range, but it's really up to you. Be precise and thorough.
- How do students collaborate successfully? Groups that split up the work typically seem to do the best. So one student worries about the survey, one about simulating and analysing data, and another about the write-up. If you're worried about using GitHub to collaborate, then just create different folders in GitHub to place your separate bits of work, and then have one person bring it together at the end.
- What intervention should we use? The intention is that you do something of interest to you. A well-written introduction would make the intervention clear.

#### **B.3.5 Rubric**

- Go/no-go #1: R is cited - [1 ‘Yes’, 0 ‘No’]
  - Both referred to in the main content and included in the reference list.
  - If not, no need to continue marking, just give paper 0 overall.
- Title - [2 ‘Exceptional’, 1 ‘Yes’, 0 ‘Poor or not done’]
  - An informative title is included.
  - Tell the reader what your story is - don’t waste their time.
  - Ideally tell them what happens at the end of the story.

- ‘Problem Set X’ is not an informative title. There should be no evidence this is a school paper.
- Author, date, and repo - [2 ‘Yes’, 0 ‘Poor or not done’]
  - The author, date of submission, and a link to a GitHub repo are clearly included. (The later likely, but not necessarily, through a statement such as: ‘Code and data supporting this analysis is available at: [LINK](#)’).
- Abstract - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - An abstract is included and appropriately pitched to a general audience.
  - The abstract answers: 1) what was done, 2) what was found, and 3) why this matters (all at a high level).
  - If your abstract is longer than four sentences then you need to think a lot about whether it is too long. It may be fine (there are always exceptions) but you should probably have a good reason.
  - Your abstract must tell the reader your top-level finding. What is the one thing that we learn about the world because of your paper?
- Introduction - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - The introduction is self-contained and tells a reader everything they need to know, including putting it into a broader context.
  - Your introduction should provide a bit of broader context to motivate the reader, as well as providing a bit more detail about what you’re interested in, what you did, what you found, why it’s important, etc.
  - A reader should be able to read only your introduction and have a good idea about the research that you carried out.
  - It would be rare that you would have tables or figures in your introduction (again there are always exceptions but think deeply about whether yours is one).
  - It must outline the structure of the paper.
  - For instance (and this is just a rough guide) an introduction for a 10 page paper, should probably be about 3 or 4 paragraphs, or 10 per cent, but it depends on specifics.
- Data - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - The survey has a clear, detailed, and justified methodology that has been thoroughly discussed. The statistical basis for the approach that is used is clear.
    - \* It would achieve the aims of the report.
    - \* Population, frame, sample, and other key aspects are described.
    - \* There is a detailed plan for reaching respondents.
    - \* Cost is discussed and appropriate.
    - \* Non-response is discussed and a plan for dealing with it clearly articulated.
    - \* There is clear respect for the survey respondents.

- When you discuss the dataset (that you have likely largely simulated) you should make sure to discuss at least:
  - \* The source of the data.
  - \* The methodology and approach that is used to collect and process the data.
  - \* The population, the frame, and the sample (as appropriate).
  - \* Information about how respondents were found. What happened to non-response?
  - \* What are the key features, strengths, and weaknesses about the survey generally.
- You should thoroughly discuss the variables in the dataset that you use. Are there any that are very similar that you nonetheless don't use? Did you construct any variables by combining various ones?
- What do the data look like?
- Plot the actual data that you're using (or as close as you can get to it).
- Discuss these plots and the other features of these data.
- These are just some of the issues strong submissions will consider. Show off your knowledge. If this becomes too detailed, then you should push some of this to footnotes or an appendix.
- ‘Exceptional’ means that when I read your submission I learn something about the dataset that I don't learn from any other submission (within a reasonable measure of course).
- Discussion - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - Some questions that a good discussion would cover include:
    - \* What is done in this paper?
    - \* What are the main points that we learn about the world?
    - \* What are some weaknesses of what was done?
    - \* What is left to learn?
  - Details of the survey - [6 ‘Exceptional’, 5 ‘Great’, 4 ‘Good’, 3 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
    - A working link to survey is included in the appendix.
    - Screenshots or at least the survey questions are included in the appendix.
      - \* The survey has been well put together.
      - \* The number and length of the questions is appropriate.
      - \* The question type and potential answers are appropriate.
      - \* It is clear how this survey could accomplish the aims of the report.
      - \* The questions flow in an appropriate way.
  - The simulation of survey responses is appropriate to the survey and the scenario - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
    - It has been done in a reproducible way and either contained in a separate R file, or is in the report appendix.
  - Numbering - [2 ‘Yes’, 0 ‘Poor or not done’]

- All figures, tables, equations, etc are numbered and referred to in the text.
- Proofreading - [2 ‘Yes’, 0 ‘Poor or not done’]
  - All aspects of submission are free of noticeable typos.
- Graphs/tables/etc - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - You must include graphs and tables in your paper and they must be to a high standard.
  - They must be well formatted and camera-ready They should be clear and digestible.
  - They must: 1) serve a clear purpose; 2) be fully self-contained through appropriate use of labels/explanations, etc; and 3) appropriately sized and coloured (or appropriate significant figures in the case of stats).
- References - [4 ‘Perfect’, 3 ‘One minor issue’, 0 ‘Poor or not done’]
  - All data/software/literature/etc are appropriately noted and cited.
  - You must cite the software and software packages that you use.
  - You must cite the datasets that you use.
  - You must cite literature that you refer to (and you should refer to literature).
  - If you take a small chunk of code from Stack Overflow then add the page in a comment next to the code.
  - If you take a large chunk of code then cite it fully.
  - 3 means one minor issue. More than one minor issue receives 0.
- Reproducibility - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - The paper and analysis must be fully reproducible.
  - A detailed README is included.
  - All code should be thoroughly documented.
  - An R project is used. Do not use `setwd()`.
  - The code must appropriately read data, prepare it, create plots, conduct analysis, generate documents, etc. Seeds are used where needed.
  - Code must have a preamble etc.
  - You must appropriately document your scripts such that someone coming in could follow them.
  - Your repo must be thoroughly organized and not contain extraneous files.
- General excellence - [3 ‘Exceptional’, 2 ‘Wow’, 1 ‘Huh, that’s interesting’, 0 ‘None’]
  - There are always students that excel in a way that is not anticipated in the rubric. This item accounts for that.

## B.4 ‘Two Cathedrals’

### B.4.1 Task

- Working individually, please conduct original research that applies methods from statistics to a question that involves an experiment.

### B.4.2 Guidance

You have various options for topics (pick one):

- Develop a research question that is of interest to you and obtain or create a relevant dataset. This option involves developing your own research question based on your own interests, background, and expertise. I encourage you to take this option, but please discuss your plans with me. How does one come up with ideas? One way is to be question-driven, where you keep an informal log of small ideas, questions, and puzzles, that you have as you’re reading and working. Often, after dwelling on it for a while you can manage to find some questions of interest. Another way is to be data-driven - try to find some interesting dataset and then work backward. Finally, yet another way, is to be methods-driven - let’s say that you happen to understand Gaussian processes, then just apply that expertise.
- A replication exercise, being sure to use the paper as a foundation rather than as a ends-in-itself.
- You should know the expectations by now. If you need a refresher then review the past problem sets. But essentially:
  - Everything is entirely reproducible.
  - Your paper must be written in R Markdown.
  - Your paper must have the following sections:
    - \* Title, date, author, keywords, abstract, introduction, data, model, results, discussion, appendix (optional, for supporting, but not critical, material), and a reference list.
  - Your paper must be well-written, draw on relevant literature, and show your statistical skills by explaining all statistical concepts that you draw on.
  - The discussion needs to be substantial. For instance, if the paper were 10 pages long then a discussion should be at least 2.5 pages. In the discussion, the paper must include subsections on weaknesses and next steps - but these must be in proportion.
  - The report must provide a link to a GitHub repo that contains everything (apart from any raw data that you git ignored if it is not yours to share). The code must be entirely reproducible, documented, and read-

able. The repo must be well-organised and appropriately use folders and README files.

#### B.4.3 Peer review submission

- My expectations for this paper are very high. I’m very excited to read what you submit. To help you achieve this standard, there is an initial ‘submission’ where you can get comments and feedback and then the final, actual, submission.
- Submit initial materials for peer-review.
  - As an individual, via Quercus, submit a PDF of your rough draft on Quercus.
  - At a minimum this must include:
    - All top-matter (title, author (you can use a pseudonym if you want), date, keywords, abstract) completely filled out.
    - A fully written Introduction section.
- All other sections must be present in your paper, but don’t have to be filled out (e.g. you must have a ‘Data’ heading, but you don’t need to have content in that section).
- To be clear - it is fine to later change any aspect of what you submit at this checkpoint.
- You will be awarded one percentage point just for submitting a draft that meets this minimum.
- The point of this is to get feedback on your work (and to make sure you have at least started thinking about this project) so you are more than welcome to (and so, if it is at all possible) include other sections that you wish to get feedback on.
- There will be no extensions granted for this submission since the following submission is dependent on this date.

#### B.4.4 Conduct peer-review

- As an individual, you will randomly be assigned a handful of rough drafts to provide feedback. You have three days to provide feedback to your peers.
- If you provide feedback to one peer you will receive one percentage point, if you provide feedback to two peers you will receive two percentage points, if you provide feedback to three (or more) peers you will receive the full three percentage points.
- Your feedback must include at least five comments (meaningful/useful bullet points). These must be well-written and thoughtful.
- There will be no extensions granted for this submission since the following submission is dependent on this date.
- Please remember that you are providing feedback here to help your colleagues. All comments should be professional and kind. It is challenging to receive criticism. Please remember that your goal here is to help your peers

advance their writing/analysis. Any feedback that is inappropriate or not up to standard will receive a 0.

#### **B.4.5 Checks**

- Do you have a causal story, or at least a sub-section in the discussion that talks about causality and why you cannot speak to it, or what you would do if you could?

#### **B.4.6 FAQ**

- Can I work as part of a team? No. It's important that you have some work that is entirely your own. You really need your own work to show off for job applications etc.
- How much should I write? Most students submit something that has 10-to-16-pages of main content, with additional pages devoted to appendices, but it's really up to you. Be precise and thorough.

#### **B.4.7 Rubric**

- Go/no-go #1: R is cited - [1 'Yes', 0 'No']
  - Both referred to in the main content and included in the reference list.
  - If not, no need to continue marking, just give paper 0 overall.
- Title - [2 'Exceptional', 1 'Yes', 0 'Poor or not done']
  - An informative title is included.
  - Tell the reader what your story is - don't waste their time.
  - Ideally tell them what happens at the end of the story.
  - 'Problem Set X' is not an informative title. There should be no evidence this is a school paper.
- Author, date, and repo - [2 'Yes', 0 'Poor or not done']
  - The author, date of submission, and a link to a GitHub repo are clearly included. (The later likely, but not necessarily, through a statement such as: 'Code and data supporting this analysis is available at: [LINK](#)').
- Abstract - [4 'Exceptional', 3 'Great', 2 'Fine', 1 'Gets job done', 0 'Poor or not done']
  - An abstract is included and appropriately pitched to a general audience.
  - The abstract answers: 1) what was done, 2) what was found, and 3) why this matters (all at a high level).
  - If your abstract is longer than four sentences then you need to think a lot about whether it is too long. It may be fine (there are always exceptions) but you should probably have a good reason.
  - Your abstract must tell the reader your top-level finding. What is the one thing that we learn about the world because of your paper?
- Introduction - [4 'Exceptional', 3 'Great', 2 'Fine', 1 'Gets job done', 0 'Poor or not done']

- The introduction is self-contained and tells a reader everything they need to know, including putting it into a broader context.
- Your introduction should provide a bit of broader context to motivate the reader, as well as providing a bit more detail about what you’re interested in, what you did, what you found, why it’s important, etc.
- A reader should be able to read only your introduction and have a good idea about the research that you carried out.
- It would be rare that you would have tables or figures in your introduction (again there are always exceptions but think deeply about whether yours is one).
- It must outline the structure of the paper.
- For instance (and this is just a rough guide) an introduction for a 10 page paper, should probably be about 3 or 4 paragraphs, or 10 per cent, but it depends on specifics.
- Data - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - You should thoroughly discuss the variables in the dataset that you use. Are there any that are very similar that you nonetheless don’t use? Did you construct any variables by combining various ones?
  - What do the data look like?
  - Plot the actual data that you’re using (or as close as you can get to it).
  - Discuss these plots and the other features of these data.
  - These are just some of the issues strong submissions will consider. Show off your knowledge. If this becomes too detailed, then you should push some of this to footnotes or an appendix.
  - ‘Exceptional’ means that when I read your submission I learn something about the dataset that I don’t learn from any other submission (within a reasonable measure of course).
- Model - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - The model is nicely written out, well-explained, justified, and appropriate.
  - When you discuss your model you must be extremely careful to spell out the statistical model that you are using defining and explaining each aspect and why it is important. Failure to do this suggests you don’t understand the model.
  - The model is appropriately complex - that is, not too simple, but not unnecessarily complicated.
  - The model has well-defined variables and these correspond to what is discussed in the data section.
  - The model needs to be written out in appropriate mathematical notation but also in plain English.
  - Every aspect of that notation must be defined otherwise the most this section can receive is poor.

- The model makes sense based on the substantive area, and also the form of the model.
- If the model is Bayesian, then priors need to be defined and sensible.
- Discussion needs to occur around how features enter the model and why. For instance, (and these are just examples) why use ages rather than age-groups, why does province have a levels effect, why is gender categorical, etc?
- In general, in order to be adequate, there needs to be a clear justification that this is the model for the situation.
- The assumptions underpinning the model are clearly discussed.
- Alternative models, or variants, must be discussed and strengths and weaknesses made clear. Why was this model chosen?
- You should mention the software that you used to run the model.
- There is some evidence of thought about the circumstances in which the model may not be appropriate.
- There is evidence of model validation and checking, whether that is out of sample or comparison to a straw man or RMSE, test/training, or appropriate sensitivity checks.
- You should be clear about model convergence, model checks, and diagnostic issues, but if this becomes too detailed then you could push some of this to an appendix.
- Great answers would discuss things such as, how do the aspects that you discussed in the data section assert themselves in the modelling decisions that you make. Again if it becomes too detailed then push some of the details to footnotes or an appendix.
- Again, explain what your model is and what is going on with it.
- Results - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - Results will likely require summary statistics, tables, graphs, images, and possibly statistical analysis or maps.
  - To be clear, you should also have text associated with all these aspects.
  - Show the reader the results by plotting them. Talk about them. Explain them. That said, this section should strictly relay results.
- Discussion - [10 ‘Exceptional’, 8 ‘Great’, 6 ‘Good’, 4 ‘Some issues’, 2 ‘Many issues’, 0 ‘Poor or not done’]
  - Some questions that a good discussion would cover include (each of these would be a sub-section of something like half a page to a page):
    - \* What is done in this this paper?
    - \* What is something that we learn about the world?
    - \* What is another thing that we learn about the world?
    - \* What are some weaknesses of what was done?
    - \* What is left to learn or how should we proceed in the future?
- Numbering - [2 ‘Yes’, 0 ‘Poor or not done’]
  - All figures, tables, equations, etc are numbered and referred to in the text.

- Proofreading - [2 ‘Yes’, 0 ‘Poor or not done’]
  - All aspects of submission are free of noticeable typos.
- Graphs/tables/etc - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - You must include graphs and tables in your paper and they must be to a high standard.
  - They must be well formatted and camera-ready They should be clear and digestible.
  - They must: 1) serve a clear purpose; 2) be fully self-contained through appropriate use of labels/explanations, etc; and 3) appropriately sized and coloured (or appropriate significant figures in the case of stats).
- References - [4 ‘Perfect’, 3 ‘One minor issue’, 0 ‘Poor or not done’]
  - All data/software/literature/etc are appropriately noted and cited.
  - You must cite the software and software packages that you use.
  - You must cite the datasets that you use.
  - You must cite literature that you refer to (and you should refer to literature).
  - If you take a small chunk of code from Stack Overflow then add the page in a comment next to the code.
  - If you take a large chunk of code then cite it fully.
  - 3 means one minor issue. More than one minor issue receives 0.
- Reproducibility - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - The paper and analysis must be fully reproducible.
  - A detailed README is included.
  - All code should be thoroughly documented.
  - An R project is used. Do not use `setwd()`.
  - The code must appropriately read data, prepare it, create plots, conduct analysis, generate documents, etc. Seeds are used where needed.
  - Code must have a preamble etc.
  - You must appropriately document your scripts such that someone coming in could follow them.
  - Your repo must be thoroughly organized and not contain extraneous files.
- Enhancements - [4 ‘Exceptional’, 3 ‘Great’, 2 ‘Fine’, 1 ‘Gets job done’, 0 ‘Poor or not done’]
  - You should pick at least one of the following and include it to enhance your submission:
    - \* Datasheets for for your datasets (see: <https://arxiv.org/abs/1803.09010>) and model cards for your models (see: <https://arxiv.org/pdf/1810.03993.pdf>).
    - \* Shiny application.
    - \* R package.
    - \* API for your model.

- There are always students that excel in a way that is not anticipated in the rubric. This item accounts for that.
- General excellence - [3 ‘Exceptional’, 2 ‘Wow’, 1 ‘Huh, that’s interesting’, 0 ‘None’]
  - There are always students that excel in a way that is not anticipated in the rubric. This item accounts for that.

#### **B.4.8 Previous examples**

Some examples of papers that well in the past include those by: Amy Farrow<sup>5</sup>, Laura Cline<sup>6</sup>, Hong Shi<sup>7</sup>, Jia Jia Ji<sup>8</sup>, and Rachael Lam<sup>9</sup>.

---

### **B.5 ‘A Proportional Response’**

#### **B.5.1 Task**

Working in teams of one to four people, please consider this scenario:

- ‘You are employed as a junior statistician at Petit Poll - a Canadian polling company. Petit Poll has a contract with a Canadian political party to provide them with monthly polling updates.’
- Working as part of a small team of 1-4 people, and in an entirely reproducible way, please write a short paper that tells the client a story about their standing.

#### **B.5.2 Recommended steps**

- Please pick a political party that you are ‘working for’, and pick a geographic focus: 1) the overall election, 2) a particular province, or 3) a specific riding.
- Then decide on a survey methodology (hint: p. 13 of Wu & Thompson provides a handy checklist). Some questions you should address include:
  - What is the population, frame, and sample?
  - What sampling methods will you use and why (e.g. you could choose SR-SWOR, stratified, etc). What are some of the statistical properties that the method brings to the table (e.g. for SRSWOR you could discuss Wu & Thompson, Theorem 2.2, etc, as appropriate)?
  - How are you going to reach your desired respondents?
  - How much do you estimate this will cost?

<sup>5</sup>[inputs/pdfs/Farrow\\_Amy-replication\\_paper.pdf](#)

<sup>6</sup>[inputs/pdfs/Cline\\_Laura-IndigenousWomenInPrison.pdf](#)

<sup>7</sup>[inputs/pdfs/Shi\\_Hong-replication\\_paper.pdf](#)

<sup>8</sup>[inputs/pdfs/Jia\\_Jia\\_Ji-effect\\_of\\_DLL\\_ON\\_spanish\\_literacy\\_skills.pdf](#)

<sup>9</sup>[inputs/pdfs/Lam\\_Rachael-final\\_paper.pdf](#)

- What steps will you take to deal with non-response and how will non-response affect your survey?
- How are you going to protect respondent privacy?
- Remember to consider all of this in the context of your ‘client’ - for instance, who would be more interested in Alberta ridings: Bloc Québécois or the Conservatives? Who likely has more money to spend - the Liberals or the Greens?
- Develop a survey on a platform that was introduced in class. Be sure to test it yourselves. You will want to test this as much as possible, maybe even swap informally with another group?
- Now release the surveys into the (simulated) ‘field’. Please do this by simulating an appropriate number of responses to your survey in R. Don’t forget to simulate in relation to the survey methodology that you proposed. Show the results and discuss your ‘findings’. Everything must be entirely reproducible. You may like to consider linking your survey ‘responses’ with other data such as the census or GSS.
- Use R Markdown to write a PDF report about all of this. Discuss your results and findings, your survey design and motivations, etc - all of it. You are writing a report that will eventually go to the ‘client’, so you must set the scene, and use language that demonstrates your command of statistical concepts but brings the reader along with you. Be sure to include graphs and tables and reference them in your discussion. Be sure to be clear about weaknesses and biases, and opportunities for future work.
- Your report must be well written. You are allowed to, and should, use mathematical notation, but you must explain all of it in plain english. Similarly, you can, and should, use surveys/sampling/observational data terminology, but again, you need to explain it.
- Your report must include at least the following aspects: title, date, authorship, non-technical executive summary, introduction, survey methodology, results, discussion, appendices that detail the survey, and references. Your ‘client’ has stats graduates working for it who need to be impressed by the main content of the report, but also has people who barely know what an average is and these people need to be impressed also. This is why your report should include a non-technical executive summary. In terms of length, this would typically be roughly 10 per cent of the report. It would be more detailed than an introduction, but still at a high level.
- Your graphs must be of an extremely high standard.
- Check that you have referenced everything. Strong submissions will draw on related literature in the discussion and would be sure to also reference those. The style of references does not matter, provided it is consistent.
- Via Quercus, submit a link to your PDF report which is hosted on GitHub. At some point in the introduction to your report, you must provide a link to the GitHub repo where the code that you used for this assignment lives (Hint: Comment. Your. Code.). Your entire workflow must be entirely reproducible.

- Please be sure to include a link to your survey in your report and screenshots of the survey in the appendix of your report.
- There should be no evidence that this is a class assignment.

### **B.5.3 Checks**

### **B.5.4 FAQ**

---

## **B.6 ‘Mr Willis of Ohio’**

### **B.6.1 Task**

- Working in teams of one to four people, and in an entirely reproducible way, please use the Canadian General Social Survey (GSS) and a regression model to tell a story.

### **B.6.2 Recommended steps**

- Depending on your focus and background, you may like to use a Bayesian hierarchical model, but regardless of the particular model that you use it must be well explained, thoroughly justified, appropriate to the task at hand, and the results must be beautifully described.
- You may focus on any year, aspect, or geography that is reasonable given the focus and constraints of the GSS. As a reminder, the GSS ‘program was designed as a series of independent, annual, cross-sectional surveys, each covering one topic in-depth.’ So please consider the topic and the year.
- The GSS is available to University of Toronto students via the library. In order to use it you need to clean and prepare it. Code to do this for one year is being distributed alongside this problem set and was discussed in lectures.
- You are welcome to simply use this code and this year, but the topic of that year will constrain your focus. Naturally, you are welcome to adapt the code to other years. If you use the code exactly as is then you must cite it. If you adapt the code then you don’t have to cite it, as it has a MIT license, but it would be appropriate to at least mention and acknowledge it, depending on how close your adaption is.
- Using R Markdown, please write a paper about your analysis and compile it into a PDF.
- Your paper must be well-written, draw on relevant literature, and show your statistical skills by explaining all statistical concepts that you draw on.
- Your paper must have the following sections: title, name/s, and date, abstract, introduction, data, model, results, discussion, and references.
- You are welcome to use appendices for supporting, but not critical, material. Your discussion must include sub-sections on weaknesses and next steps.
- In your report you must provide a link to a GitHub repo that fully contains

your analysis. Your code must be entirely reproducible, documented, and readable. Your repo must be well-organised and appropriately use folders.

- Your graphs and tables must be of an incredibly high standard. Graphs and tables should be well formatted and report-ready. They should be clean and digestible. Furthermore, you should label and describe each table/figure.
- When you discuss the dataset (in the data section) you should make sure to discuss (at least):
  - Its key features, strengths, and weaknesses generally.
  - A discussion of the questionnaire - what is good and bad about it?
  - A discussion of the methodology including how they find people to take the survey; what their population, frame, and sample were; what sampling approach they took and what some of the trade-offs may be; what they do about non-response; the cost.
  - This is just some of the issues strong submissions will consider. Show off your knowledge. If this becomes too detailed then you should push some of this to footnotes or an appendix.
- When you discuss your model (in the model section), you must be extremely careful to spell out the statistical model that you are using, defining and explaining each aspect and why it is important. (For a Bayesian model, a discussion of priors and regularization is almost always important.) You should mention the software that you used to run the model. You should be clear about model convergence, model checks, and diagnostic issues. How do the sampling and survey aspects that you discussed assert themselves in the modelling decisions that you make? Again, if it becomes too detailed then push some of the details to footnotes or an appendix.
- You should present model results, graphs, figures, etc, in the results section. This section should strictly relay results. Interpretation of these results and conclusions drawn from the results should be left for the discussion section.
- Your discussion should focus on your model results. Interpret them and explain what they mean. Put them in context. What do we learn about the world having understood your model and its results? What caveats could apply? To what extent does your model represent the small world and the large world (to use the language of McElreath, Ch 2)? What are some weaknesses and opportunities for future work?
- Check that you have referenced everything. Strong submissions will draw on related literature in the discussion (and other sections) and would be sure to also reference those. The style of references does not matter, provided it is consistent.
- As a team, via Quercus, submit a PDF of your paper. Again, in your paper you must have a link to the associated GitHub repo in an appendix. And you must include the R Markdown file that produced the PDF in that repo.
- A good way to work as a team would be to split up the work, so that one person is doing each section. The people doing the sections that rely on data (such as the analysis and the graphs) could just simulate it while they are waiting for the person putting together the data to finish.

- It is expected that your submission be well written and able to be understood by the average reader of say 538. This means that you are allowed to use mathematical notation, but you must be able to explain it all in plain English. Similarly, you can (and hint: you should) use survey, sampling, observational, and statistical terminology, but again you need to explain it. Your work should have flow and should be easy to follow and understand. To communicate well, anyone at the university level should be able to read your report once and relay back the methodology, overall results, findings, weaknesses and next steps without confusion.

### **B.6.3 Checks**

- It is recommended that you (informally) proofread one another's sections - why not exchange papers with another group?
- Everyone in the team receives the same mark.
- There should be no evidence that this is a class assignment.

### **B.6.4 FAQ**

---

## **B.7 ‘Five Votes Down’**

### **B.7.1 Task**

- The primary goal of this paper is to predict the overall popular vote of the 2020 American presidential election using multilevel regression with post-stratification.

### **B.7.2 Recommended steps**

- We expect you to work as part of a group of 4 people, but groups of size 1-4 are fine. We have suggested a split of the work based on a 4-person group, but these are just suggestions.
- Individual-level survey data:
  - Request access to the Democracy Fund + UCLA Nationscape ‘Full Data Set’: <https://www.voterstudygroup.org/publication/nationscape-data-set>. This could take a day or two. Please start early.
  - Given the expense of collecting this data, and the privilege of having access to it, if you don’t properly cite this dataset then you will get zero for this problem set.
  - Once you have access then pick a survey of interest. We will use “ns20200102.dta” in the example (your number may be different).
  - This will be a large file and is not yours to share. Do not push it to GitHub (use the .gitignore file - see here: <https://carpentries-incubator.github.io/git-Rstudio-course/02-ignore/index.html>).

- Use the example R code to get started preparing this dataset, and then go on cleaning and preparing it based on what you need.
- Make graphs and tables about the survey data and write beautiful sentences and paragraphs explaining everything.
- Post-stratification data:
  - We will use the American Community Surveys (ACS).
  - Please create an account with IPUMS: <https://usa.ipums.org/usa/index.shtml>
  - You want the 2018 1-year ACS. Then you need to select some variables. This will depend on what you want to model and the survey data, but some options include: REGION, STATEICP, AGE, SEX, MARST, RACE, HISPAN, BPL, CITIZEN, EDUC, LABFORCE, INC-TOT. Have a look around and see what you are interested in, remembering that you will need to establish a correspondence to the survey.
  - Download the relevant post-stratification data (it’s probably easiest to change the data format to .dta). Again, this can take some time. Please start this early.
  - This will be a large file and is not yours to share. Do not push it to GitHub (use the .gitignore file - see here: <https://carpentries-incubator.github.io/git-Rstudio-course/02-ignore/index.html>).
  - Given the expense of collecting this data, and the privilege of having access to it, if you don’t properly cite this dataset then you will get zero for this problem set.
  - Clean and prepare the post-stratification dataset.
  - Remember that you need cell counts for the sub-populations in your model. See examples in the readings.
- (It may be efficient to start with simulated data while waiting for the real data) Modelling.
  - You will want to explain vote intention based on a variety of explanatory variables. Construct the vote intention variable so that it is binary (either ‘supports Trump’ or ‘supports Biden’).
  - You are welcome to use lm() but you would need to explain the nuances of this decision in the model section (Hint: start here: <https://statmodeling.stat.columbia.edu/2020/01/10/linear-or-logistic-regression-with-binary-outcomes/>).
  - That said, you should probably use logistic regression if it is at all possible for you. If you don’t know where to start then look at (in increasing levels of complexity) glm(), lme4::glmer(), or brms::brm(). There are examples of each in the readings.
  - Think very deeply about model fit, diagnostics, and other similar things that you need in order to convince someone that your model is appropriate.
  - You have flexibility of the model that you use, (and hence the cells that you will need to create next). In general, the more cells the better, but

you may want fewer cells for simplicity in the writing process and to ensure a decent sample in each cell.

- Apply your trained model to the post-stratification dataset to make the best estimate of the election result that you can. The specifics will depend on your modelling approach but will likely involve `predict()`, `add_predicted_draws()`, or similar. See the examples in the readings. We are primarily interested in the distribution of your forecast of the overall Presidential popular vote, and how the explanatory variables affect this. But great submissions would go beyond that. Also, you're taking a statistics course, so if you just gave a central estimate and nothing else, then that would not be great.
- Create beautiful graphs and tables of your model and results.
- Create wonderful paragraphs talking about and explaining everything.
- (Again, it's probably efficient to start with simulated data/results while waiting)
  - Write up.
  - Using R Markdown, please write a very thorough paper about your analysis and compile it into a PDF.
  - The paper must be well-written, draw on relevant literature, and show your statistical skills by explaining all statistical concepts that you draw on.
  - The paper must have the following sections: title, name/s, and date, abstract and keywords, introduction, data, model, results, discussion, and references.
  - The paper may use appendices for supporting, but not critical, material.
  - The discussion needs to be substantial. For instance, if the paper were 10 pages long then a discussion should be at least 2.5 pages. In the discussion, the paper must include subsections on weaknesses and next steps - but these must be in proportion.
  - The report must provide a link to a GitHub repo that contains everything (apart from the raw data that you git ignored because it is not yours to share). The code must be entirely reproducible, documented, and readable. The repo must be well-organised and appropriately use folders and README files.
  - The graphs and tables must be of an incredibly high standard, well formatted, and report-ready. They should be clean and digestible. Furthermore, you should label and describe each table/figure.
  - When you discuss the datasets (in the data section) (remember there will be at least two datasets to discuss) you should make sure to discuss (at least):
    - \* Their key features, strengths, and weaknesses generally.
    - \* The survey questionnaire - what is good and bad about it?
    - \* A discussion of the methodology including how they find people to take the survey; what their population, frame, and sample were;

what sampling approach they took and what some of the trade-offs may be; what they do about non-response; the cost.

- \* This is just some of the issues strong submissions will consider. Show off your knowledge. If this becomes too detailed then you should push some of this to footnotes or an appendix.
- The dataset section is probably an appropriate place to include an explanation of what post-stratification is (in non-statistical language) and the strengths and weaknesses of it, although this discussion may fit more naturally in another section. Regardless, be sure to justify the inclusion of each explanatory variable.
- When you discuss your model (in the model section), you must be extremely careful to spell out the statistical model that you are using, defining and explaining each aspect and why it is important. (For a Bayesian model, a discussion of priors and regularization is almost always important.) You should mention the software that you used to run the model. You should be clear about model convergence, model checks, and diagnostic issues, although you may push the details of this to an appendix depending on how detailed you get. How do the sampling and survey aspects that you discussed assert themselves in the modelling decisions that you make? How can you convince a reader that you have neither overfit nor underfit the data? Again, if it becomes too detailed then push some of the details to footnotes or an appendix.
- You should present model results, graphs, figures, etc, in the results section. This section should strictly relay results. It must include text explaining all of these and summary statistics and similar. However, interpretation of these results and conclusions drawn from the results should be left for the discussion section.
- Your discussion should focus on your model results, but this time interpreting them, and explaining what they mean. Put them in context. What do we learn about the world having understood your model and its results? What caveats could apply? To what extent does your model represent the small world and the large world (to use the language of McElreath, Ch 2)? What are some weaknesses and opportunities for future work? Who is going to win the election? How confident are you in that forecast? Do you have a small or large distribution? What could that mean? Are you more confident in certain states? Do certain explanatory variables carry more weight than others? Etc.
- Check that you have referenced everything. Strong submissions will draw on related literature in the discussion (and other sections) and would be sure to also reference those. The style of references does not matter, but it must be consistent.
- If you don’t cite R then you will get zero for this problem set.
- As a team, via Quercus, submit a PDF of your paper. Again, in your paper you must have a link to the associated GitHub repo. And you must include the R Markdown file that produced the PDF in that repo. The

RMarkdown file must exactly produce the PDF. Don't edit it manually ex post - that isn't reproducible.

- A good way to work as a team would be to split up the work, so that one person is doing each section. The people doing the sections that rely on data (such as the analysis and the graphs) could just simulate it while they are waiting for the person putting together the data to finish. We have recommended a split above, but you do what works for you.

### **B.7.3 Checks**

- It is expected that your submission be well written and able to be understood by the average reader of say 538. This means that you are allowed to use mathematical notation, but you must be able to explain it all in plain English. Similarly, you can (and hint: you should) use survey, sampling, observational, and statistical terminology, but again you need to explain it. The average person doesn't know what a p-value is nor what a confidence interval is. You need to explain all of this in plain language the first time you use it. Your work should have flow and should be easy to follow and understand. To communicate well, anyone at the university level should be able to read your report once and relay back the methodology, overall results, findings, weaknesses and next steps without confusion.
- It is recommended that you (informally) proofread one another's work - why not exchange papers with another group?
- Everyone in the team receives the same mark.
- There should be no evidence that this is a class assignment.

### **B.7.4 FAQ**

---

## **B.8 ‘What’s next?’**

### **B.8.1 Task**

Please work individually. In this paper, you will conduct original research that applies methods from statistics to a question involving surveys, sampling or observational data.

### **B.8.2 Recommended steps**

You have various options for topics (pick one): Develop a research question that is of interest to you and obtain or create a relevant dataset. This option involves developing your own research question based on your own interests, background, and expertise. I encourage you to take this option, but please discuss your plans with me. How does one come up with ideas? One way is to be question-driven, where you keep an informal log of

small ideas, questions, and puzzles, that you have as you’re reading and working. Often, after dwelling on it for a while you can manage to find some questions of interest. Another way is to be data-driven - try to find some interesting dataset and then work backward. Finally, yet another way, is to be methods-driven - let’s say that you happen to understand Gaussian processes, then just apply that expertise. (Thanks to Jack Bailey for this idea.) Build a MRP model based on the CES and a post-stratification dataset that you obtain, to identify how the 2019 Canadian Federal Election would have been different if ‘everyone’ had voted. What do we learn about the importance of turnout based on your model and results? (This option involves logistic regression in either frequentist or Bayesian settings.) Reproduce a paper. This means that you download the data and then write your own code (using their code and paper as a guide if it’s available) to try to get their results and then write up what you did and found. Options include: Angelucci, Charles, and Julia Cagé, 2019, ‘Newspapers in times of low advertising revenues’, American Economic Journal: Microeconomics, please see: <https://www.openicpsr.org/openicpsr/project/116438/version/V1/view>. (This option can be accomplished with just OLS. It is a ‘safe’ pick. I even already provided you with some code in class to get started - see the notes! ). Bailey, Michael A., Daniel J. Hopkins & Todd Rogers, 2016, ‘Unresponsive and Unpersuaded: The Unintended Consequences of a Voter Persuasion Effort’, Political Behavior. Clark, Sam, 2019, ‘A General Age-Specific Mortality Model With an Example Indexed by Child Mortality or Both Child and Adult Mortality’, Demography, please see: <https://github.com/sinafala/svd-comp>. (This is an ambiguous pick!) Skinner, Ben, 2019, ‘Making the connection: Broadband access and online course enrollment at public open admissions institutions’, Research in Higher Education, please see: [https://github.com/btskinner/oa\\_online\\_broadband\\_rep](https://github.com/btskinner/oa_online_broadband_rep). Pons, Vincent, 2018, ‘Will a Five-Minute Discussion Change Your Mind? A Countrywide Experiment on Voter Choice in France’ American Economic Review. Valencia Caicedo, Felipe, 2019, ‘The Mission: Human Capital Transmission, Economic Persistence, and Culture in South America’, The Quarterly Journal of Economics, please see: <https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/ML1155>. If you have a favourite paper and want to reproduce it, then please let me know by the end of Week 12 so that I can check that it’s appropriate. Pretend that you work for Upworthy. Request the Upworthy dataset and then use it to evaluate the result of an A/B test. This request could take a week. Please plan ahead if you choose this option. Critique the following paper: AlShebli, Bedoor, Kinga Makovi & Talal Rahwan, 2020, ‘The association between early career informal mentorship in academic collaborations and junior author performance’, Nature Communications. You should be able to download the data here: <https://github.com/bedoor/Mentorship> and the paper here: <https://www.nature.com/articles/s41467-020-19723-8>. For background and starting points for your critique, please see:

<https://statmodeling.stat.columbia.edu/2020/11/19/are-female-scientists-worse-mentors-this-study-pretends-to-know/> and <https://danieleweeks.github.io/Mentorship/#summary>. (This option involves extensive data exploration and thinking really hard about what they are trying to do and how they are doing it.) Use this post by Andrew Whitby - <https://andrewwhitby.com/2020/11/24/contact-tracing-biased/> - as a starting point to explore biased sampling and its effects on what we know about COVID and how this affects public policy. (This option involves extensive simulation.) In ‘Bias Behind Bars’, data journalist Tom Cardoso finds that ‘(a)fter controlling for a number of variables, ... Black and Indigenous inmates are more likely to get worse scores than white inmates, based solely on their race.’ The main story is here: <https://www.theglobeandmail.com/canada/article-investigation-racial-bias-in-canadian-prison-risk-assessments/> The methodology discussion is here: <https://www.theglobeandmail.com/canada/article-investigation-racial-bias-in-canadian-prisons-methodology/> The observational data is available here: [https://www.theglobeandmail.com/files/editorial/News/nw-na-risk-1023/The\\_Globe\\_and\\_Mail\\_CSC\\_OMS\\_2012-2018\\_20201022235635.zip](https://www.theglobeandmail.com/files/editorial/News/nw-na-risk-1023/The_Globe_and_Mail_CSC_OMS_2012-2018_20201022235635.zip)

Your task is to follow the methodology that Tom published and attempt to replicate the results. Are you able to replicate them? Do the results change significantly under slightly different assumptions? (This option involves only frequentist logistic regression, although doing everything in a Bayesian setting would be lovely too). You should know the expectations by now. If you need a refresher then review the past problem sets. Everything is entirely reproducible. Your paper must be written in R Markdown. Your paper must have the following sections: Title, date, author, keywords, abstract, introduction, data, model, results, discussion, appendix (optional, for supporting, but not critical, material), and a reference list. Your paper must be well-written, draw on relevant literature, and show your statistical skills by explaining all statistical concepts that you draw on. The discussion needs to be substantial. For instance, if the paper were 10 pages long then a discussion should be at least 2.5 pages. In the discussion, the paper must include subsections on weaknesses and next steps - but these must be in proportion. The report must provide a link to a GitHub repo that contains everything (apart from any raw data that you git ignored if it is not yours to share). The code must be entirely reproducible, documented, and readable. The repo must be well-organised and appropriately use folders and README files. My expectations for this paper are very high. I’m very excited to read what you submit. To help you achieve this standard, there are two initial ‘submissions’ where you can get comments and feedback and then the final, actual, submission. Due dates: (Optional) December 9 11:59pm ET Submit initial materials for peer-review. As an individual, via Quercus, submit a PDF of your rough draft on Quercus by 11:59pm ET on Wednesday, December 9, 2020. At a minimum this must include: All top-matter (title, author (you can use a pseudonym if you want), date, keywords, abstract) completely filled

out. A fully written Introduction section. All other sections must be present in your paper, but don’t have to be filled out (e.g. you must have a ‘Data’ heading, but you don’t need to have content in that section). To be clear - it is fine to later change any aspect of what you submit at this checkpoint. You will be awarded 1 percentage point just for submitting a draft that meets this minimum (that is 1 out of the 30 that are available for the final paper). If you don’t submit, then the percentage point will be pushed to part d). The point of this is to get feedback on your work (and to make sure you have at least started thinking about this project) so you are more than welcome to include other sections that you wish to get feedback on. There will be no extensions granted for this submission since the following submission is dependent on this date. (Optional) December 12 11:59pm ET Conduct peer-review. As an individual, on December 10, you will randomly be assigned a handful of rough drafts to provide feedback. You have until December 12, 2020 11:59pm ET to provide feedback to your peers. If you provide feedback to one peer you will receive 1 percentage point, if you provide feedback to two peers you will receive 2 percentage points, if you provide feedback to three (or more) peers you will receive the full 3 percentage points. You may complete this aspect whether or not you submitted something in part a). If you don’t complete it then the percentage points will be pushed to part d). Your feedback must include at least six comments (meaningful/useful bullet points). These must be well-written and thoughtful. There will be no extensions granted for this submission since the following submission is dependent on this date. Please remember that you are providing feedback here to help your colleagues. All comments should be professional and kind. It is challenging to receive criticism. Please remember that your goal here is to help your peers advance their writing/analysis. Any feedback that is inappropriate or not up to standard will receive a 0 and cannot be redeemed later. (Optional) December 16 11:59pm ET Submit materials for TA review. Submit a PDF to Quercus. The TA will provide high-level comments on December 17. At a minimum this must include: All top-matter. Fully written Introduction, Data, Model, and Results sections. All other sections must be present in your paper, but don’t have to be filled out (e.g. you must have a ‘Discussion’ heading, but you don’t need to have content in that section). To be clear - it is fine to later change any aspect of what you submit at this checkpoint. You receive 1 percentage point for submitting something that meets that minimum. If you don’t submit anything then this is pushed to the final paper. There are no extensions possible on this aspect. (Compulsory) December 20 11:59pm ET As an individual, via Quercus, submit a PDF of your paper. Again, in your paper, you must have a link to the associated GitHub repo. This submission will be graded based on a rubric that will be posted on Quercus and will be worth 25-30 percentage points depending on parts a) - c).

**B.8.3 Checks****B.8.4 FAQ**

- Do I have to submit an initial paper in order to do the peer-review? Yes.

---

## Bibliography

---

- Abeysooriya, M., Soria, M., Kasu, M. S., and Ziemann, M. (2021). Gene name errors: Lessons not learned. *PLOS Computational Biology*, 17(7):1–13.
- Acemoglu, D., Johnson, S., and Robinson, J. A. (2001). The colonial origins of comparative development: An empirical investigation. *American Economic Review*, 91(5):1369–1401.
- Akerlof, G. A. (1978). The market for ‘lemons’: Quality uncertainty and the market mechanism. In *Uncertainty in economics*, pages 235–251. Elsevier.
- Alexander, M. (2019a). Analyzing name changes after marriage using a non-representative survey. <https://www.monicaalexander.com/posts/2019-08-07-mrp/>.
- Alexander, M. (2019b). The concentration and uniqueness of baby names in australia and the us. <https://www.monicaalexander.com/posts/2019-20-01-babynames/>.
- Alexander, M. (2019c). Reproducibility in demographic research. Technical report.
- Alexander, M. (2021). Overcoming barriers to sharing code. *YouTube*. <https://youtu.be/yvM2C6aZ94k>.
- Alexander, M. and Alkema, L. (2018). Global estimation of neonatal mortality using a bayesian hierarchical splines regression model. *Demographic Research*, 38:335–372.
- Alexander, M. J., Kiang, M. V., and Barbieri, M. (2018). Trends in black and white opioid mortality in the united states, 1979–2015. *Epidemiology (Cambridge, Mass.)*, 29(5):707.
- Alexander, R. and Alexander, M. (2021). The increased effect of elections and changing prime ministers on topics discussed in the australian federal parliament between 1901 and 2018.
- Alexander, R., Caetano, S.-J., Chen, H., Chong, M., Collins, A., Deng, S., Ehrlich, I., Hodgetts, P., Joo, Y., Pejcinovska, M., Walaa, M., and Wankiewicz, M. (2021). An introduction to dosstoolkit.
- Alexander, R. and Hodgetts, P. A. (2021). *AustralianPoliticians: Provides Datasets About Australian Politicians*. R package version 0.1.0.

- Allaire, J., Iannone, R., Presmanes Hill, A., and Xie, Y. (2021). *distill: 'R Markdown' Format for Scientific and Technical Writing*. <https://github.com/rstudio/distill>, <https://pkgs.rstudio.com/distill>.
- Allen, E. J., Dechow, P. M., Pope, D. G., and Wu, G. (2017). Reference-dependent preferences: Evidence from marathon runners. *Management Science*, 63(6):1657–1672.
- Allen, J. (2021). *plumberDeploy: Plumber Deployment*. R package version 0.2.1.
- Alsan, M. and Wanamaker, M. (2018). Tuskegee and the health of black men. *The quarterly journal of economics*, 133(1):407–455.
- Arel-Bundock, V. (2021a). *modelsummary: Summary Tables and Plots for Statistical Models and Data: Beautiful, Customizable, and Publication-Ready*. R package version 0.9.4.
- Arel-Bundock, V. (2021b). *WDI: World Development Indicators and Other World Bank Data*. R package version 2.7.4.
- Arnold, J. B. (2021). *ggthemes: Extra Themes, Scales and Geoms for 'ggplot2'*. R package version 4.2.4.
- Association, A. M. and of Medicine, N. Y. A. (1848). *Code of medical ethics*. Academy of Medicine.
- Athey, S. and Imbens, G. W. (2017). The state of applied econometrics: Causality and policy evaluation. *Journal of Economic Perspectives*, 31(2):3–32.
- Au, R. (2020). Data cleaning is analysis, not grunt work. Technical report, Counting Stuff.
- Bandy, J. and Vincent, N. (2021). Addressing "documentation debt" in machine learning research: A retrospective datasheet for bookcorpus.
- Barrett, M. (2021a). *Data science as an atomic habit*. 16 January.
- Barrett, M. (2021b). *ggdag: Analyze and Create Elegant Directed Acyclic Graphs*. R package version 0.2.3.
- Barron, A. T., Huang, J., Spang, R. L., and DeDeo, S. (2018). Individuals, institutions, and innovation in the debates of the french revolution. *Proceedings of the National Academy of Sciences*, 115(18):4607–4612.
- Bastian, H. (2020). A timeline of the oxford-astrazeneca covid-19 vaccine trials. Technical report.
- Bates, D., Mächler, M., Bolker, B., and Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, 67(1):1–48.

- Baumer, B., Kaplan, D., and Horton, N. (2021). *Modern Data Science With R*. CRC Press, 2 edition.
- Beauregard, K. and Sheppard, J. (2021). Antiwomen but proquota: Disaggregating sexism and support for gender quota policies. *Political Psychology*, 42(2):219–237.
- Bensinger, G. (2020). *Google redraws the borders on maps depending on who's looking*. <https://www.washingtonpost.com/technology/2020/02/14/google-maps-political-borders/>.
- Berkson, J. (1946). Limitations of the application of fourfold table analysis to hospital data. *Biometrics Bulletin*, 2(3):47–53.
- Bertrand, M. and Mullainathan, S. (2004). Are emily and greg more employable than lakiisha and jamal? a field experiment on labor market discrimination. *American economic review*, 94(4):991–1013.
- Blair, G., Cooper, J., Coppock, A., and Humphreys, M. (2019). Declaring and diagnosing research designs. *American Political Science Review*, 113:838–859.
- Bland, J. M. and Altman, D. (1986). Statistical methods for assessing agreement between two methods of clinical measurement. *The lancet*, 327(8476):307–310.
- Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, 55(4):77–84.
- Blei, D. M. and Lafferty, J. D. (2009). Topic models. In *Text Mining*, pages 101–124. Chapman and Hall/CRC.
- Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022.
- Bloom, H., Bell, A., and Reiman, K. (2020). Using data from randomized trials to assess the likely generalizability of educational treatment-effect estimates from regression discontinuity designs. *Journal of Research on Educational Effectiveness*, pages 1–30.
- Boland, P. J. (1984). A biographical glimpse of william sealy gosset. *The American Statistician*, 38(3):179–183.
- Bowers, J. (2011). Six steps to a better relationship with your future self. *The Political Methodologist*, 18(2):2–8.
- Bowley, A. L. (1901). *Elements of Statistics*. P. S. King.
- Brandt, A. M. (1978). Racism and research: the case of the tuskegee syphilis study. *Hastings center report*, pages 21–29.

- Briggs, R. C. (2021). Why does aid not target the poorest? *International Studies Quarterly*, 65(3):739–752.
- Brokowski, C. and Adli, M. (2019). Crispr ethics: moral considerations for applications of a powerful tool. *Journal of molecular biology*, 431(1):88–101.
- Bronte, C. (1847). *Jane Eyre*.
- Brontë, C. (1857). *The Professor*.
- Brook, R. H., Ware, J. E., Rogers, W. H., Keeler, E. B., Davies, A. R., Sherbourne, C. D., Goldberg, G. A., Lohr, K. N., Camp, P., and Newhouse, J. P. (1984). The effect of coinsurance on the health of adults: results from the rand health insurance experiment.
- Bryan, J. (2020). *Happy Git and GitHub for the useR*.
- Bryan, J. and Hester, J. (2020). *What They Forgot to Teach You About R*.
- Bryan, J., Hester, J., Robinson, D., and Wickham, H. (2019). *reprex: Prepare Reproducible Example Code via the Clipboard*. R package version 0.3.0.
- Buhr, R. (2017). *Using R as a Production Machine Learning Language (Part I)*. 17 October.
- Buja, A., Cook, D., and Swayne, D. F. (1996). Interactive high-dimensional data visualization. *Journal of computational and graphical statistics*, 5(1):78–99.
- Cahill, N., Weinberger, M., and Alkema, L. (2020). What increase in modern contraceptive use is needed in fp2020 countries to reach 75% demand satisfied by 2030? an assessment using the accelerated transition method and family planning estimation model. *Gates Open Research*, 4.
- Cambon, J. and Belanger, C. (2021). tidygeocoder: Geocoding made easy. R package version 1.0.2.
- Carle, E. (1969). *The Very Hungry Caterpillar*. World Publishing Company.
- Caro, R. (2019). *Working*. Knopf, 1 edition.
- Carroll, L. (1865). *Alice's Adventures in Wonderland*. Macmillan.
- Carroll, L. (1871). *Through the Looking-Glass*. Macmillan.
- Chamberlain, S., Wickham, H., and Chang, W. (2021). *analogsea: Interface to 'Digital Ocean'*. <https://github.com/sckott/analogsea> (devel) <https://analogsea.icu/> (docs).
- Chambliss, D. F. (1989). The mundanity of excellence: An ethnographic report on stratification and olympic swimmers. *Sociological theory*, 7(1):70–86.

- Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., Allen, J., McPherson, J., Dipert, A., and Borges, B. (2021). *shiny: Web Application Framework for R*. R package version 1.6.0.
- Chen, W., Chen, X., Hsieh, C.-T., and Song, Z. (2019). A forensic examination of china's national accounts. Technical report, National Bureau of Economic Research.
- Cheng, J., Karambelkar, B., and Xie, Y. (2021). *leaflet: Create Interactive Web Maps with the JavaScript 'Leaflet' Library*. R package version 2.0.4.1.
- Choudechova, A., Benavides-Prado, D., Fialko, O., and Vaithianathan, R. (2018). A case study of algorithm-assisted decision making in child maltreatment hotline screening decisions. In Friedler, S. A. and Wilson, C., editors, *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pages 134–148. PMLR.
- Chrétien, J. (2007). *My Years as Prime Minister*. Knopf Canada.
- City of Toronto (2021). *2021 Street Needs Assessment*. <https://www.toronto.ca/city-government/data-research-maps/research-reports/housing-and-homelessness-research-and-reports/>.
- Cleveland, W. (1994). *The Elements of Graphing Data*. Hobart Press, 2 edition.
- Cohn, A. (2019). Data and code for: Civic Honesty Around the Globe.
- Cohn, A., Maréchal, M. A., Tannenbaum, D., and Zünd, C. L. (2019a). Civic honesty around the globe. *Science*, 365(6448):70–73.
- Cohn, A., Maréchal, M. A., Tannenbaum, D., and Zünd, C. L. (2019b). Supplementary materials for: Civic honesty around the globe. *Science*, 365(6448):70–73.
- Cohn, N. (2016). *We Gave Four Good Pollsters the Same Raw Data. They Had Four Different Results*. The New York Times, The Upshot, 20 September, <https://www.nytimes.com/interactive/2016/09/20/upshot/the-error-the-polling-world-rarely-talks-about.html>.
- Cook, D., Reid, N., and Tanaka, E. (2021). The foundation is available for thinking about data visualization inferentially. *Harvard Data Science Review*. <https://hdsr.mitpress.mit.edu/pub/mpdasaqt>.
- Cooksey, B. (2014). An introduction to apis.
- Cooley, D. (2020). *mapdeck: Interactive Maps Using 'Mapbox GL JS' and 'Deck.gl'*. R package version 0.3.4.
- Cox, M. (2021). Inside Airbnb - Toronto Data.

- Craiu, R. V. (2019). The hiring gambit: In search of the twofer data scientist. *Harvard Data Science Review*, 1(1).
- Crawford, K. (2021). *Atlas of AI*. Yale University Press.
- Cunningham, S. (2021). *Causal Inference: The Mixtape*. Yale Press.
- Dagan, N., Barda, N., Kepten, E., Miron, O., Perchik, S., Katz, M. A., Hernán, M. A., Lipsitch, M., Reis, B., and Balicer, R. D. (2021). Bnt162b2 mrna covid-19 vaccine in a nationwide mass vaccination setting. *New England Journal of Medicine*.
- Dahly, D. (2020). *A brief history of medical statistics and its impact on reproducibility*. 10 August.
- Darling, W. M. (2011). A theoretical and practical implementation tutorial on topic modeling and gibbs sampling. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 642–647.
- DeWitt, H. (2000). *The Last Samurai*. Talk Mirimax Books.
- D’Ignazio, C. and Klein, L. F. (2020). *Data feminism*. Mit Press.
- Doll, R. and Hill, A. B. (1950). Smoking and carcinoma of the lung. *British medical journal*, 2(4682):739.
- Edgeworth, F. Y. (1885). Methods of statistics. *Journal of the Statistical Society of London*, pages 181–217.
- Eghbal, N. (2020). *Working in public: the making and maintenance of open source software*. Stripe Press.
- Euller, R., Long, S. H., and Marquis, M. S. (1997). Data Cleaning Procedures for the 1993 Robert Wood Johnson Foundation Employer Health Insurance Survey. Technical report, RAND CORP SANTA MONICA CA.
- Farrugia, P., Petrisor, B. A., Farrokhyar, F., and Bhandari, M. (2010). Research questions, hypotheses and objectives. *Canadian journal of surgery*, 53(4):278.
- Finkelstein, A., Taubman, S., Wright, B., Bernstein, M., Gruber, J., Newhouse, J. P., Allen, H., Baicker, K., and Group, O. H. S. (2012). The oregon health insurance experiment: evidence from the first year. *The Quarterly journal of economics*, 127(3):1057–1106.
- Firke, S. (2020). *janitor: Simple Tools for Examining and Cleaning Dirty Data*. R package version 1.2.1.
- Fisher, R. (1935). *The Design of Experiments*. Oliver and Boyd.
- Fiske, S. T. and Kuriwaki, S. (2021). Words to the wise on writing scientific papers.

- Fitts, A. S. (2014). The king of content: How upworthy aims to alter the web, and could end up altering the world.
- Foer, J. (2011). *Moonwalking with Einstein*. Penguin Books.
- Forster, E. M. (1927). *Aspects of the Novel*. Edward Arnold.
- Fourcade, M. and Healy, K. (2017). Seeing like a market. *Socio-economic review*, 15(1):9–29.
- Fox, J. and Andersen, R. (2006). Effect displays for multinomial and proportional-odds logit models. *Sociological Methodology*, 36(1):225–255.
- Franconeri, S. L., Padilla, L. M., Shah, P., Zacks, J. M., and Hullman, J. (2021). The science of visual data communication: What works. *Psychological Science in the Public Interest*, 22(3):110–161.
- Franklin, L. R. (2005). Exploratory experiments. *Philosophy of Science*, 72(5):888–899.
- Friedman, J. H., Tibshirani, R., and Hastie, T. (2009). *The Elements of Statistical Learning*. Springer.
- Friendly, M. and Wainer, H. (2021). *A History of Data Visualization and Graphic Communication*. Harvard University Press, 1 edition.
- Gagolewski, M. (2020). *R package stringi: Character string processing facilities*.
- Galef, J. (2020). Episode 248: Are democrats being irrational? (david shor). *Rationally Speaking*. <http://rationallyspeakingpodcast.org/248-are-democrats-being-irrational-david-shor/>.
- Gao, Y., Kennedy, L., Simpson, D., Gelman, A., et al. (2021). Improving multilevel regression and poststratification with structured priors. *Bayesian Analysis*.
- Garnier, Simon, Ross, Noam, Rudis, Robert, Camargo, Pedro, A., Sciaiani, Marco, Scherer, and Cédric (2021). *viridis - Colorblind-Friendly Color Maps for R*. R package version 0.6.2.
- Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., au2, H. D. I., and Crawford, K. (2020). Datasheets for datasets.
- Gelfand, S. (2020). *opendatatoronto: Access the City of Toronto Open Data Portal*. R package version 0.1.4.
- Gelfand, S. (2021). Make a reprex... please. YouTube. <https://youtu.be/G5Nm-GpmrLw>.
- Gelman, A. (2016). What has happened down here is the winds have changed. Technical report.

- Gelman, A. (2020). Statistical models of election outcomes. *YouTube*. <https://youtu.be/7gjDnrbLQ4k>.
- Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., and Rubin, D. (2014). *Bayesian Data Analysis*. CRC Press, 3 edition.
- Gelman, A. and Hill, J. (2007). *Data Analysis Using Regression and Multi-level/Hierarchical Models*.
- Gelman, A., Hill, J., and Vehtari, A. (2020). *Regression and Other Stories*. Cambridge University Press.
- Gelman, A. and Loken, E. (2013). The garden of forking paths: Why multiple comparisons can be a problem, even when there is no “fishing expedition” or “p-hacking” and the research hypothesis was posited ahead of time. *Department of Statistics, Columbia University*, 348.
- Gelman, A., Mattson, G., and Simpson, D. (2018). Gaydar and the fallacy of decontextualized measurement. *Sociological Science*, 5(12):270–280.
- Gelman, A., Pasarica, C., and Dodhia, R. (2002). Let’s practice what we preach: turning tables into graphs. *The American Statistician*, 56(2):121–130.
- Gelman, A. and Vehtari, A. (2020). What are the most important statistical ideas of the past 50 years? *arXiv preprint arXiv:2012.00174*.
- Gerber, A. and Green, D. (2012). *Field Experiments: Design, Analysis, and Interpretation*. W W Norton.
- Gertler, P. J., Martinez, S., Premand, P., Rawlings, L. B., and Vermeersch, C. M. (2016). *Impact evaluation in practice*. The World Bank.
- Ghitza, Y. and Gelman, A. (2020). Voter registration databases and mrp: Toward the use of large-scale databases in public opinion research. *Political Analysis*, 28(4):507–531.
- Goodrich, B., Gabry, J., Ali, I., and Brilleman, S. (2020). rstanarm: Bayesian applied regression modeling via Stan. R package version 2.21.1.
- Graham, P. (2020). *How to write usefully*. February, <http://paulgraham.com/useful.html>.
- Greenland, S., Senn, S. J., Rothman, K. J., Carlin, J. B., Poole, C., Goodman, S. N., and Altman, D. G. (2016). Statistical tests, p values, confidence intervals, and power: a guide to misinterpretations. *European journal of epidemiology*, 31(4):337–350.
- Griffiths, T. and Steyvers, M. (2004). Finding scientific topics. *PNAS*, 101:5228–5235.

- Grolemund, G. and Wickham, H. (2011). Dates and times made easy with lubridate. *Journal of Statistical Software*, 40(3):1–25.
- Grün, B. and Hornik, K. (2011). topicmodels: An R package for fitting topic models. *Journal of Statistical Software*, 40(13):1–30.
- Halberstam, D. (1972). *The Best and the Brightest*. Random House.
- Hamming, R. W. (1996). *The Art of Doing Science and Engineering*. Stripe Press.
- Hanretty, C. (2020). An introduction to multilevel regression and post-stratification for estimating constituency opinion. *Political Studies Review*, 18(4):630–645.
- Hao, K. (2019). This is how AI bias really happens—and why it’s so hard to fix. *MIT Technology Review*.
- Hart, E. M., Barmby, P., LeBauer, D., Michonneau, F., Mount, S., Mulrooney, P., Poisot, T., Woo, K. H., Zimmerman, N. B., and Hollister, J. W. (2016). Ten simple rules for digital data storage.
- Hastie, T. J. and Tibshirani, R. J. (1990). *Generalized additive models*, volume 43. CRC press.
- Hayot, E. (2014). *The Elements of Academic Style*. Columbia University Press.
- Healy, K. (2018). *Data Visualization*. Princeton University Press.
- Healy, K. (2020). *The Kitchen Counter Observatory*.
- Heckman, J. J. (1979). Sample selection bias as a specification error. *Econometrica: Journal of the econometric society*, pages 153–161.
- Heil, B. J., Hoffman, M. M., Markowitz, F., Lee, S.-I., Greene, C. S., and Hicks, S. C. (2021). Reproducibility standards for machine learning in the life sciences. *Nature Methods*, 18(10):1132–1135.
- Henry, L. and Wickham, H. (2020). *purrr: Functional Programming Tools*. R package version 0.3.4.
- Hernan, M. A. and Robins, J. M. (2020). *What If*. CRC Press.
- Hillel, W. (2017). *How Do We Trust Our Science Code?* 14 August, <https://www.hillelwayne.com/how-do-we-trust-science-code/>.
- Hug, L., Alexander, M., You, D., Alkema, L., and for Child, U. I.-a. G. (2019). National, regional, and global levels and trends in neonatal mortality between 1990 and 2017, with scenario-based projections to 2030: a systematic analysis. *The Lancet Global Health*, 7(6):e710–e720.

- Hugh-Jones, D. (2020). *huxtable: Easily Create and Style Tables for LaTeX, HTML and Other Formats*. R package version 5.1.1.
- Hughes, N. and Rutter, J. (2016). *Oliver Letwin*. 15 December 2016, <https://www.instituteforgovernment.org.uk/ministers-reflect/person/oliver-letwin/>.
- Hulley, S. B. (2007). *Designing clinical research*. Lippincott Williams & Wilkins.
- Hullman, J. and Gelman, A. (2021). Designing for interactive exploratory data analysis requires theories of graphical inference. *Harvard Data Science Review*. <https://hdsr.mitpress.mit.edu/pub/w075glo6>.
- Huntington-Klein, N. (2021). *The Effect: An Introduction to Research Design and Causality*. Chapman & Hall.
- Huntington-Klein, N., Arenas, A., Beam, E., Bertoni, M., Bloem, J., Burli, P. H., Chen, N., Grieco, P. L., Ekpe, G., Pugatch, T., et al. (2020). The influence of hidden researcher decisions in applied microeconomics.
- Huntington-Klein, N., Arenas, A., Beam, E., Bertoni, M., Bloem, J. R., Burli, P., Chen, N., Grieco, P., Ekpe, G., Pugatch, T., et al. (2021). The influence of hidden researcher decisions in applied microeconomics. *Economic Inquiry*.
- Iannone, R. (2020). *DiagrammeR: Graph/Network Visualization*. R package version 1.0.6.1.
- Iannone, R., Cheng, J., and Schloerke, B. (2020). *gt: Easily Create Presentation-Ready Display Tables*. R package version 0.2.2.
- Igelström, E. (2020). Causal graphs in r with diagrammer.
- Ilyas, I. F. and Chu, X. (2019). *Data cleaning*. Morgan & Claypool.
- Imai, K. (2017). *Quantitative Social Science*. Princeton University Press.
- Ishiguro, K. (1989). *The Remains of the Day*. Faber and Faber.
- James, G., Witten, D., Hastie, T., and Tibshirani, R. (2017). *An Introduction to Statistical Learning with Applications in R*.
- Johnson, A. A., Ott, M., and Dogucu, M. (2022). *Bayes Rules! An Introduction to Bayesian Modeling with R*. CRC Press.
- Jones, A. H. (1953). Census records of the later roman empire. *The Journal of Roman Studies*, 43(1-2):49–64.
- Jordan, M. I. (2019). Artificial intelligence—the revolution hasn't happened yet. *Harvard Data Science Review*, 1(1). <https://hdsr.mitpress.mit.edu/pub/wot7mkc1>.

- Joyner, M. J. (1991). Modeling: optimal marathon performance on the basis of physiological factors. *Journal of applied physiology*, 70(2):683–687.
- Kahle, D. and Wickham, H. (2013). ggmap: Spatial visualization with ggplot2. *The R Journal*, 5(1):144–161.
- Kahneman, D., Sibony, O., and Sunstein, C. (2021). *Noise: A Flaw in Human Judgment*. William Collins.
- Kastellec, J., Lax, J., and Phillips, J. (2016). Estimating state public opinion with multi-level regression and poststratification using r.
- Kay, M. (2020). *tidybayes: Tidy Data and Geoms for Bayesian Models*. R package version 2.3.1.
- Kearney, M. W. (2019). rtweet: Collecting and analyzing twitter data. *Journal of Open Source Software*, 4(42):1829. R package version 0.7.0.
- Kennedy, L. and Gabry, J. (2020). Mrp with rstanarm. As accessed 28 April 2021.
- Kennedy, L. and Gelman, A. (2020). Know your population and know your model: Using model-based regression and poststratification to generalize findings beyond the observed sample.
- Kennedy, L., Khanna, K., Simpson, D., and Gelman, A. (2020). Using sex and gender in survey adjustment.
- Kenny, C. T., Kuriwaki, S., McCartan, C., Rosenman, E., Simko, T., and Imai, K. (2021). The impact of the u.s. census disclosure avoidance system on redistricting and voting rights analysis.
- Keyes, O. (2019). Counting the countless. Technical report.
- Kharecha, P. A. and Hansen, J. E. (2013). Prevented mortality and greenhouse gas emissions from historical and projected nuclear power. *Environmental science & technology*, 47(9):4889–4895.
- Kiang, M. V., Tsai, A. C., Alexander, M. J., Rehkoppf, D. H., and Basu, S. (2021). Racial/ethnic disparities in opioid-related mortality in the usa, 1999–2019: the extreme case of washington dc. *Journal of Urban Health*, 98(5):589–595.
- Kimmerer, R. W. (2012). *Braiding Sweetgrass*. Milkweed Editions.
- King, G. (2006). Publication, publication. *PS: Political Science & Politics*, 39(1):119–125.
- King, S. (2000). *On Writing: A Memoir of the Craft*. Scribner.
- Kleiber, C. and Zeileis, A. (2008). *Applied Econometrics with R*. Springer-Verlag, New York. ISBN 978-0-387-77316-2.

- Koenker, R. and Zeileis, A. (2009). On reproducible econometric research. *Journal of Applied Econometrics*, 24(5):833–847.
- Kohavi, R., Tang, D., and Xu, Y. (2020). *Trustworthy Online Controlled Experiments: A Practical Guide to A/B Testing*. Cambridge University Press.
- Kross, S. (2021). *postcards: Create Beautiful, Simple Personal Websites*. R package version 0.2.0.
- Kuhn, M. and Wickham, H. (2020). *Tidymodels: a collection of packages for modeling and machine learning using tidyverse principles*.
- Kuznets, S. (1941). *National Income and Its Composition, 1919-1938*. National Bureau of Economic Research.
- Lamott, A. (1994). *Bird by Bird: Some instructions on writing and life*. Anchor Books.
- Larmarange, J. (2021). *labelled: Manipulating Labelled Data*. R package version 2.9.0.
- Lauderdale, B. E., Bailey, D., Blumenau, J., and Rivers, D. (2020). Model-based pre-election polling for national and sub-national outcomes in the us and uk. *International Journal of Forecasting*, 36(2):399–413.
- Lazear, E. P. (2000). Economic imperialism. *the Quarterly Journal of economics*, 115(1):99–146.
- Lee, B. D. (2018). Ten simple rules for documenting scientific software.
- Leek, J. and Peng, R. D. (2020). Advanced Data Science 2020.
- Leeper, T. J. (2018). *tabulizer: Bindings for Tabula PDF Table Extractor Library*. R package version 0.2.2.
- Levitt, S. D. (1997). Using electoral cycles in police hiring to estimate the effect of police on crime. *The American Economic Review*, 87(3).
- Levitt, S. D. (2002). Using electoral cycles in police hiring to estimate the effects of police on crime: Reply. *American Economic Review*, 92(4):1244–1250.
- Lin, S., Ali, I., and Wilson, G. (2020). Ten quick tips for making things findable. *PLoS Computational Biology*, 16(12):e1008469.
- Locke, S. and D’Agostino McGowan, L. (2018). *datasauRus: Datasets from the Datasaurus Dozen*. R package version 0.1.4.
- Lohr, S. L. (2019). *Sampling: Design and Analysis*. CRC Press.
- Lopp, S. (2017). R for enterprise: Understanding r’s startup.

- Lovelace, R., Nowosad, J., and Muenchow, J. (2019). *Geocomputation with R*. CRC Press.
- Lucas Jr, R. E. (1978). Asset prices in an exchange economy. *Econometrica: journal of the Econometric Society*, pages 1429–1445.
- Luebke, D. M. and Milton, S. (1994). Locating the victim: An overview of census-taking, tabulation technology, and persecution in nazi germany. *IEEE Annals of the History of Computing*, 16(3):25.
- Lumley, T. (2020). survey: analysis of complex survey samples. R package version 4.0.
- Lüdecke, D., Makowski, D., Waggoner, P., and Patil, I. (2020). performance: Assessment of regression models performance. CRAN. R package.
- MacDorman, M. F. and Declercq, E. (2018). The failure of united states maternal mortality reporting and its impact on women's lives. *Birth (Berkeley, Calif.)*, 45(2):105.
- Martinez, L. R. (2019). How much should we trust the dictator's gdp growth estimates? Available at SSRN 3093296.
- Matias, J. N., Munger, K., Quere, M. A. L., and Ebersole, C. (2019). The upworthy research archive. Technical report.
- Mattson, G. (2017). Artificial intelligence discovers gayface. sigh. Technical report.
- McCrory, J. (2002). Using electoral cycles in police hiring to estimate the effect of police on crime: Comment. *American Economic Review*, 92(4):1236–1243.
- McElreath, R. (2020). *Statistical Rethinking: A Bayesian Course with Examples in R and Stan*. CRC Press.
- McPhee, J. (2017). *Draft No. 4*. Farrar, Straus and Giroux.
- McQuire, S. (2019). One map to rule them all? google maps as digital technical object. *Communication and the Public*, 4(2):150–165.
- Meng, X.-L. (2018). Statistical paradises and paradoxes in big data (i): Law of large populations, big data paradox, and the 2016 us presidential election. *The Annals of Applied Statistics*, 12(2):685–726.
- Merali, Z. (2010). Computational science:... error. *Nature*, 467(7317):775–777.
- Michener, W. K. (2015). Ten simple rules for creating a good data management plan. *PLoS computational biology*, 11(10):e1004525.
- Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., and Gebru, T. (2019). Model cards for model reporting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*.

- Miyakawa, T. (2020). No raw data, no science: another possible source of the reproducibility crisis.
- Mock, T. (2020). *Building a blog with distill.* <https://themockup.blog/posts/2020-08-01-building-a-blog-with-distill/>.
- Murphy, H. (2017). *Why Stanford Researchers Tried to Create a 'Gaydar' Machine.* The New York Times, 9 October, <https://www.nytimes.com/2017/10/09/science/stanford-sexual-orientation-study.html>.
- Müller, K. (2017). *here: A Simpler Way to Find Your Files.* R package version 0.1.
- Nelder, J. A. and Wedderburn, R. W. (1972). Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384.
- Neufeld, M. J. (2002). Wernher von braun, the ss, and concentration camp labor: Questions of moral, political, and criminal responsibility. *German Studies Review*, 25(1):57–78.
- Neuwirth, E. (2014). *RColorBrewer: ColorBrewer Palettes.* R package version 1.1-2.
- Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. (2019). Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453.
- OECD (2014). The essential macroeconomic aggregates. In *Understanding National Accounts*, pages 13–46. OECD.
- OECD (2022). *Quarterly GDP.* doi: 10.1787/b86d1fc8-en.
- Ooms, J. (2019a). *pdftools: Text Extraction, Rendering and Converting of PDF Documents.* R package version 2.3.
- Ooms, J. (2019b). *pdftools: Text Extraction, Rendering and Converting of PDF Documents.* R package version 2.3.
- Ooms, J. (2019c). *tesseract: Open Source OCR Engine.* R package version 4.1.
- Ostrom, T. (2021). Funding of clinical trials and reported drug efficacy.
- Oreopoulos, P. and Petronijevic, U. (2018). Student coaching: How far can technology go? *Journal of Human Resources*, 53(2):299–329.
- Orwell, G. (1946). *Politics and the English Language.* <https://www.orwellfoundation.com/the-orwell-foundation/orwell/essays-and-other-works/politics-and-the-english-language/>.

- Oxford-AstraZeneca (2020). Azd1222 vaccine met primary efficacy endpoint in preventing covid-19. Technical report.
- Pavlik, K. (2019). Understanding + classifying genres using spotify audio features.
- Pedersen, T. L. (2020). *patchwork: The Composer of Plots*. R package version 1.1.1.
- Phillips, A. W. (1958). The relation between unemployment and the rate of change of money wage rates in the united kingdom, 1861-1957. *economica*, 25(100):283–299.
- Pineau, J., Vincent-Lamarre, P., Sinha, K., Larivière, V., Beygelzimer, A., d’Alché Buc, F., Fox, E., and Larochelle, H. (2021). Improving reproducibility in machine learning research: a report from the neurips 2019 reproducibility program. *Journal of Machine Learning Research*, 22.
- Pitman, J. (1993). *Probability*.
- Presmanes Hill, A. (2021a). *M-F-E-O: postcards + distill*. As accessed 9 May 2021.
- Presmanes Hill, A. (2021b). *Up & running with blogdown in 2021*. <https://alison.rbind.io/post/new-year-new-blogdown/>.
- R Core Team (2021). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.
- Register, Y. (2020). Data science ethics in 6 minutes.
- Rilke, R. M. (1929). *Letters to a Young Poet*.
- Robinson, D., Hayes, A., and Couch, S. (2020). *broom: Convert Statistical Objects into Tidy Tibbles*. R package version 0.7.0.
- Robinson, E. and Nolis, J. (2020). *Build a Career in Data Science*.
- Rockoff, H. (2019). On the controversies behind the origins of the federal economic statistics. *Journal of Economic Perspectives*, 33(1):147–64.
- Rudis, B. (2020). *hrbrthemes: Additional Themes, Theme Components and Utilities for 'ggplot2'*. R package version 0.8.0.
- Ruggles, S., Fitch, C., Magnuson, D., and Schroeder, J. (2019). Differential privacy and census data: Implications for social and economic research. In *AEA papers and proceedings*, volume 109, pages 403–08.
- Salganik, M. (2018). *Bit by bit: Social Research in the Digital Age*. Princeton University Press.
- Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 3(3):210–229.

- Schloerke, B., Allaire, J., Borges, B., and Aden-Buie, G. (2021). *learnr: Interactive Tutorials for R*. <https://rstudio.github.io/learnr/>, <https://github.com/rstudio/learnr>.
- Schloerke, B. and Allen, J. (2021). *plumber: An API Generator for R*. R package version 1.1.0.
- Si, Y. (2020). On the use of auxiliary variables in multilevel regression and poststratification.
- Silberzahn, R., Uhlmann, E. L., Martin, D. P., Anselmi, P., Aust, F., Awtrey, E., Bahnik, Š., Bai, F., Bannard, C., Bonnier, E., et al. (2018). Many analysts, one data set: Making transparent how variations in analytic choices affect results. *Advances in Methods and Practices in Psychological Science*, 1(3):337–356.
- Silge, J. (2018). *Text classification with tidy data principles*.
- Silver, N. (2020). *We Fixed An Issue With How Our Primary Forecast Was Calculating Candidates' Demographic Strengths*. FiveThirtyEight, 19 February, <https://fivethirtyeight.com/features/we-fixed-a-mistake-in-how-our-primary-forecast-was-calculating-candidates-demographic-strengths/>.
- Simpson, E. H. (1951). The interpretation of interaction in contingency tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 13(2):238–241.
- Sjoberg, D. D., Curry, M., Hannum, M., Larmarange, J., Whiting, K., and Zabor, E. C. (2021). *gtsummary: Presentation-Ready Data Summary and Analytic Result Tables*. R package version 1.4.0.
- Slowikowski, K. (2021). *ggrepel: Automatically Position Non-Overlapping Text Labels with 'ggplot2'*. R package version 0.9.1.
- Staniak, M. and Biecek, P. (2019). The landscape of r packages for automated exploratory data analysis. *arXiv preprint arXiv:1904.02101*.
- Statistics Canada (2017). Guide to the census of population, 2016. Technical report, Statistics Canada.
- Statistics Canada (2020). Sex at birth and gender: Technical report on changes for the 2021 census. Technical report, Statistics Canada.
- Steyvers, M. and Griffiths, T. (2006). Probabilistic topic models. In Landauer, T., McNamara, D., Dennis, S., and Kintsch, W., editors, *Latent Semantic Analysis: A Road to Meaning*.
- Stock, J. H. and Trebbi, F. (2003). Retrospectives: Who invented instrumental variable regression? *Journal of Economic Perspectives*, 17(3):177–194.
- Student (1908). The probable error of a mean. *Biometrika*, pages 1–25.

- Sunstein, C. R. and Reisch, L. A. (2017). *The economics of nudge*. Routledge.
- Suriyakumar, V. M., Papernot, N., Goldenberg, A., and Ghassemi, M. (2020). Chasing your long tails: Differentially private prediction in health care settings.
- Taback, N. (2020). *Design of Experiments and Observational Studies*.
- Taddy, M. (2019). *Business Data Science*. McGraw Hill.
- The Economist (2013). *Johnson: Those six little rules: George Orwell on writing*. 29 July, <https://www.economist.com/prospero/2013/07/29/johnson-those-six-little-rules>.
- Thompson, C., Parry, J., Phipps, D., and Wolff, T. (2020). *spotifyr: R Wrapper for the 'Spotify' Web API*. R package version 2.1.1.
- Thornhill, J. (2021). Lunch with the ft: Mathematician hannah fry. *Financial Times*.
- Tierney, N. (2017). visdat: Visualising whole data frames. *JOSS*, 2(16):355.
- Tierney, N. (2020). *R Markdown for Scientists*. <https://rmd4sci.njtierney.com>.
- Timbers, T.-A., Campbell, T., and Lee, M. (2022). *Data Science: A First Introduction*. CRC Press.
- Tolley, E. and Paquet, M. (2021). Gender, municipal party politics, and montreal's first woman mayor. *Canadian Journal of Urban Research*, 30(1):40–52.
- Tukey, J. W. (1962). The future of data analysis. *The annals of mathematical statistics*, 33(1):1–67.
- UN IGME (2021). Levels and trends in child mortality, 2021.
- Van den Broeck, J., Argeseanu Cunningham, S., Eeckels, R., and Herbst, K. (2005). Data cleaning: detecting, diagnosing, and editing data abnormalities. *PLoS medicine*, 2(10):e267.
- Vanderplas, S., Cook, D., and Hofmann, H. (2020). Testing statistical charts: What makes a good graph? *Annual Review of Statistics and Its Application*, 7:61–88.
- von Bergmann, J., Shkolnik, D., and Jacobs, A. (2021). *cancensus: R package to access, retrieve, and work with Canadian Census data and geography*. R package version 0.4.2.
- Wang, W., Rothschild, D., Goel, S., and Gelman, A. (2015). Forecasting elections with non-representative polls. *International Journal of Forecasting*, 31(3):980–991.
- Wang, Y. and Kosinski, M. (2018). Deep neural networks are more accurate

- than humans at detecting sexual orientation from facial images. *Journal of personality and social psychology*, 114(2):246.
- Wardrop, R. L. (1995). Simpson's paradox and the hot hand in basketball. *The American Statistician*, 49(1):24–28.
- Ware, J. (1989). Investigating therapies of potentially great benefit: Ecmo. *Statistical Science*, (4):298–306.
- Wasserman, L. (2005). *All of Statistics*. Springer.
- Weissgerber, T. L., Milic, N. M., Winham, S. J., and Garovic, V. D. (2015). Beyond bar and line graphs: time for a new data presentation paradigm. *PLoS biology*, 13(4):e1002128.
- Whitby, A. (2020). *The Sum of the People*. Basic Books.
- WHO (2019). Trends in maternal mortality 2000 to 2017: estimates by who, unicef, unfpfa, world bank group and the united nations population division.
- Wickham, H. (2010). A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, 19(1):3–28.
- Wickham, H. (2014). Tidy data. *Journal of statistical software*, 59(1):1–23.
- Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.
- Wickham, H. (2017). *tidyverse: Easily Install and Load the 'Tidyverse'*. R package version 1.2.1.
- Wickham, H. (2019a). *Advanced R*. CRC Press.
- Wickham, H. (2019b). *babynames: US Baby Names 1880-2017*. R package version 1.0.0.
- Wickham, H. (2019c). *httr: Tools for Working with URLs and HTTP*. R package version 1.4.1.
- Wickham, H. (2019d). *rvest: Easily Harvest (Scrape) Web Pages*. R package version 0.3.5.
- Wickham, H. (2019e). *stringr: Simple, Consistent Wrappers for Common String Operations*. R package version 1.4.0.
- Wickham, H. (2020a). *forcats: Tools for Working with Categorical Variables (Factors)*. R package version 0.5.0.
- Wickham, H. (2020b). *Tidyverse*.
- Wickham, H. (2021a). *Mastering shiny*.
- Wickham, H. (2021b). *tidyr: Tidy Messy Data*. R package version 1.1.3.

- Wickham, H. (2021c). *The tidyverse style guide*. <https://style.tidyverse.org/index.html>.
- Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019a). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.
- Wickham, H., Averick, M., Bryan, J., Chang, W., McGowan, L. D., François, R., Grolemund, G., Hayes, A., Henry, L., Hester, J., Kuhn, M., Pedersen, T. L., Miller, E., Bache, S. M., Müller, K., Ooms, J., Robinson, D., Seidel, D. P., Spinu, V., Takahashi, K., Vaughan, D., Wilke, C., Woo, K., and Yutani, H. (2019b). Welcome to the tidyverse. *Journal of Open Source Software*, 4(43):1686.
- Wickham, H. and Bryan, J. (2020). *usethis: Automate Package and Project Setup*. R package version 1.6.1.
- Wickham, H., François, R., Henry, L., and Müller, K. (2020a). *dplyr: A Grammar of Data Manipulation*. R package version 0.8.5.
- Wickham, H. and Grolemund, G. (2017). *R for Data Science*.
- Wickham, H., Hester, J., and Bryan, J. (2021). *readr: Read Rectangular Text Data*. R package version 2.1.1.
- Wickham, H., Hester, J., and Chang, W. (2020b). *devtools: Tools to Make Developing R Packages Easier*. R package version 2.3.2.
- Wickham, H. and Miller, E. (2020). *haven: Import and Export 'SPSS', 'Stata' and 'SAS' Files*. R package version 2.3.1.
- Wiessner, P. W. (2014). Embers of society: Firelight talk among the ju/'hoansi bushmen. *Proceedings of the National Academy of Sciences*, 111(39):14027–14035.
- Wilk, M. B. and Gnanadesikan, R. (1968). Probability plotting methods for the analysis for the analysis of data. *Biometrika*, 55(1):1–17.
- Wilkinson, L. (2005). *The Grammar of Graphics*. Springer, 2 edition.
- Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., et al. (2016). The fair guiding principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9.
- Wilson, G. (2021). *Building Software Together*. CRC Books.
- Wilson, G., Bryan, J., Cranston, K., Kitzes, J., Nederbragt, L., and Teal,

- T. K. (2017). Good enough practices in scientific computing. *PLOS Computational Biology*, 13(6):1–20.
- Wright, P. G. (1928). *The Tariff on Animal and Vegetable Oils*. Macmillan Company.
- Wu, C. and Thompson, M. E. (2020). *Sampling Theory and Practice*. Springer.
- Xie, Y. (2021). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.37.
- Xie, Y., Dervieux, C., and Hill, A. P. (2021a). *blogdown: Create Blogs and Websites with R Markdown*. R package version 1.2.
- Xie, Y., Thomas, A., and Presmanes Hill, A. (2021b). *blogdown: Creating Websites with R Markdown*. <https://bookdown.org/yihui/blogdown/>.
- Zhu, H. (2020). *kableExtra: Construct Complex Table with 'kable' and Pipe Syntax*. R package version 1.3.1.
- Zinsser, W. (1976). *On Writing Well*.
- Zook, M., Barcas, S., Boyd, D., Crawford, K., Keller, E., Gangadharan, S. P., Goodman, A., Hollander, R., Koenig, B. A., Metcalf, J., et al. (2017). Ten simple rules for responsible big data research.