PUBLISHED BY PROJECT MOSAIC BOOKS

TUFTE-LATEX.GOOGLECODE.COM

Cover inset photo: Richard Marx. Cover photo: Lichens on the North Shore, by the author.

First Corrected Printing, November 2015

Preface

Information is what we want, but data are what we've got. The techniques for transforming data into information go back hundreds of years. A good starting marker is 1592 with the publication of weekly "bills of mortality" in London. These bills were tabulations: a condensing of the data into a form more readily assimilated by the human reader. Constructing such tabulations was a manual operation.

As data became larger, machines were introduced to speed up the tabulations. A major step was Hollerith's development of punched cards and an electrical tabulating system for the US Census of 1890. This was so successful that Hollerith started a company, IBM, that came to play an important role in the development of today's electronic computers.

Also in the late 19th century, statistical methods began to develop rapidly. These methods have been tremendously important in interpreting data, but they were not intrinsically tied to mechanical data processing. Generations of students have learned to carry out statistical operations by hand on small sets of data, typically from a laboratory experiment.

Nowadays, it's common to have datasets that are so large they can be processed only by machine. In this era of "big data," often data are amassed by networks of instruments and computers. The settings for this are diverse: the genome, satellite observations of Earth, entries by web users, sales transactions, etc. With such data, there are new opportunities for finding and characterizing patterns using techniques such as data mining, machine learning, data visualization, and so on. As a result, everyday uses of data involve computer processing of data. This includes data cleaning, combining data from multiple sources, and transformation into a form suitable for input to data-condensing operations like visualization.

In writing this book I hope to help people gain the understanding and skills for data wrangling, a process of preparing data for visualization and other modern techniques of statistical interpretation. Doing so inevitably involves, at the center, using the computer in a sophisticated way.

Need sophisticated computing require extended study of computer programming? My view is that it does not. First, over the last half century, a coherent set of simple data operations have been developed that can be used as building blocks of sophisticated data wrangling processes. The trick is not mastering programming but learning to think in terms of these operations. Much of this book is intended to help you master such thinking. Second, it's possible to use recent developments in software to reduce vastly the amount of programming needed to use the data operations. I've drawn on such software — particularly R and the packages `dplyr` and `ggplot2` developed by Hadley Wickham and the many others who contribute to important open-source projects — to focus on a small subset of functions that accomplish data wrangling tasks in a concise and expressive way. The computer notation is clear enough that, once you have learned to read it, constructing new data wrangling programs can be done by modifying working examples. (Experienced R programmers will note the distinctive style of R statements in this book, including a consistent focus on a small set of key notation.)

I've tried to arrange this book to be accessible to a reader with no technical computing background. The book derives from notes for a course at Macalester College, *Data Computing Fundamentals*, that has no pre-requisites. To be effective, the book should be read along with practice using the computer. Some of this practice involves becoming familiar with the user-interface software. This includes the RStudio environment for computing with R, the use of a text editor, and RMarkdown, the notation that integrates the computer commands with graphical display and explanatory narrative. It's hard to write effective explanations of how to work with user-interfaces; doing it is the best way to learn, watching others do it is also helpful. If you're using this book in a classroom setting, your instructor can provide the demonstrations. If you are reading this book on your own, you can make use of the many videos that introduce using RStudio, its editor, and the RMarkdown document-writing system.

The contributions of many people are behind this book. At a start, this includes the work of the people in the R Core Team and the fantastic community of open-source R developers. Of particular importance to *Data Computing* are Hadley Wickham, Winston Chang, Joe Cheng, Garrett Grolemund, and JJ Allaire at RStudio. Professors Nicholas Horton, Randall Pruim, Elizabeth Shoop and Paul Overvoorde helped in defining the Data Computing Fundamentals course. Collaborators on Project MOSAIC helped encourage the *minimal-R* approach that let's you do a lot with a little bit of R. The many faculty participants in the Computing and Visualization Consortium of small liberal arts colleges provided important feedback to identify

gaps and shortcomings. The careful attention of Jeff Witmer, Craig Ching, Michael Schneider, and Dennis McGuire pointed out many deficiencies in the manuscript. Students in the Data Computing Fundamentals course at Macalester College willingly hiked the initial rough path of the course curriculum. Macalester students Barbara Borges Ribiero, Mengdie Wang, and Jingjing Yang were teaching assistants for the course and spent summer months developing case studies, visualizations, and interactive "apps" for displaying what data verbs do. Andrew Zieffler, Ethan Brown, and Erik Anderson from the University of Minnesota helped in sharpening assessment, developing case studies, and teaching an early version of the course. Prof. Jessen Havill at Denison University put together a Mellon Foundation workshop in 2010 to discuss how to bring computing into the science curriculum; the ideas of a short course and a focus on data emerged from that workshop.

Financial support for the Data Computing Fundamentals course and the Computation and Visualization Consortium was generously provided by the Howard Hughes Medical Institute. The National Science Foundation supported Project MOSAIC (NSF DUE-0920350). Both were instrumental to developing the Macalester course and this book.

My dear wife, Maya, and our beloved daughters Tamar, Liat, and Netta were encouraging (and patient) during many dinner-table discussions of case studies and the many hours I was away writing notes for the Data Computing course and teaching the Wednesday-evening class sessions.

Daniel Kaplan
St. Paul, Minnesota
August 2015

Contents

1
Tidy Data

Data can be as simple as a column of numbers in a spreadsheet file or as complex as the medical records collected by a hospital. A newcomer to working with data may expect each source of data to be organized in a unique way and to require unique techniques. The expert, however, has learned to operate with a small, standard set of tools. As you'll see, each of the standard tools performs a comparatively simple task. Combining those simple tasks in appropriate ways is the key to dealing with complex data.

One of the reason the individual tools can be simple is this: each tool gets applied to data arranged in a simple but precisely defined pattern called *tidy data*. At the heart of tidy data is the *data table*. To illustrate, Table 1.1 a handful of entries from a large US Social Security Administration tabulation of names given to babies. In particular, the table shows how many babies of each sex were given each name in each year.

TIDY DATA: A format for data that provides a systematic approach to computer processing. You will be using tidy data throughout this book and, likely, for the large majority of your professional work with data.

DATA TABLE: A rectangular array of data.

name	sex	count	year
Sherina	F	19	1970
Leketha	F	6	1973
Tahirih	F	6	1976
Radha	F	13	1979
Hatim	M	8	1981
Cissy	F	7	1984
Ahmet	M	6	1986
... and so on for 1,792,091 rows			

Table 1.1: A data table showing how many babies were given each name in each year in the US.

Table 1.1 shows there were 6 boys named Ahmet born in the US in 1986 and 19 girls named Sherina born in 1970. As a whole, the `BabyNames` data table covers the years 1880 through 2013 and includes a total of 333,417,770 individuals, somewhat larger than the current population of the US.

The data in Table 1.1 are "tidy" because they are organized according to two simple rules.

1. The rows, called *case*s, each refer to a specific, unique and similar sort of thing, e.g. girls named Sherina in 1970.
2. The columns, called *variable*s, each have the same sort of value recorded for each row. For instance, `count` gives the number of babies for each case; `sex` tells which gender the case refers to.

CASE: The individual people, objects, events, etc. from which variables are measured.

VARIABLE: The attributes of each case that are measured or observed.

When data are in tidy form, it's relatively straightforward to transform the data into arrangements that are more useful for answering interesting questions. For instance, you might wish to know which

were the most popular baby names over all the years. Even though Table 1.1 contains the popularity information implicitly, some re-arrangement is needed (for instance, adding up the counts for a name across all the years) before the information is made explicit, as in Table 1.2.

The process of transforming information that is implicit in a data table into another data table that gives the information explicitly is called *data wrangling*. The wrangling itself is accomplished by using *data verbs* that take a tidy data table and transform it into another tidy data table in a different form. In the following chapters, you will be introduced to the various *data verbs*.

sex	name	total_births
M	James	5091189
M	John	5073958
M	Robert	4789776
M	Michael	4293460
F	Mary	4112464
... and so on for 102,690 rows		

Table 1.2: The most popular baby names across all years.

	A	B	E	F	G	H	I	J
1			City of Minneapolis Statistics					
2			General Election November 5, 2013					
3	Ward	Precinct	Voters Registering by Absentee	Total Registrations	Voters at Polls	Absentee Voters	Total Ballots Cast	Total Turnout
4	City-Wide Total		708	6,634	75,145	4,954	80,099	33.38%
5								
6	1	1	3	28	492	27	519	27.23%
7	1	2	1	44	836	56	892	31.71%
8	1	3	0	40	905	19	924	38.87%
9	1	4	5	29	768	26	794	36.62%
10	1	5	0	31	683	31	714	37.46%
11	1	6	0	69	739	20	759	32.62%
12	1	7	0	47	291	8	299	15.79%
13	1	8	0	43	415	5	420	30.55%
14	1	9	0	42	596	25	621	25.42%
15	Ward 1 Subtotal		9	373	5,725	217	5,942	30.93%
16								
17	2	1	1	63	1,011	39	1,050	36.42%
18	2	2	5	44	679	37	716	50.39%
19	2	3	4	48	324	18	342	18.88%
20	2	4	0	53	117	3	120	7.34%
21	2	5	2	50	495	26	521	25.49%
22	2	6	1	36	433	19	452	39.10%
23	2	7	0	39	138	7	145	13.78%
24	2	8	1	50	1,206	36	1,242	47.90%
25	2	9	2	39	351	16	367	30.56%
26	2	10	0	87	196	5	201	6.91%
27	Ward 2 Subtotal		16	509	4,950	206	5,156	27.56%
28								
29	3	1	0	52	165	1	166	7.04%

Table 1.3: Ward and precinct votes cast in the 2013 Minneapolis mayoral election.

Table 1.3, in contrast to `BabyNames`, is not in tidy form. True, table 1.3 is attractive and neatly laid out. There are helpful labels and summaries that make it easy for a person to read and draw conclusions. (For instance, Ward 1 had a higher voter turnout than Ward 2, and both wards were lower than the city total.)

Being neat is not what makes data tidy. Table 1.3, however neat it is, violates the rules for tidy data.

- Rule 1: The rows, called *cases*, each must represent the same underlying attribute, that is, the same kind of thing.

  That's not true in Table 1.3. For most of the table, the rows represent a single precinct. But other rows give ward or city-wide totals. The first two rows are captions describing the data, not cases.

- Rule 2: Each column is a variable containing the same type of

value for each case.

That's mostly true in Table 1.3, but the tidy pattern is interrupted by labels that are not variables. For instance, the first two cells in row 15 are the label "Ward 1 Subtotal," different from the ward/precinct identifiers that are the values in most of the first column.

Conforming to the rules for tidy data simplifies summarizing and analyzing data. For instance, in the tidy baby names table, it's easy to find the total number of babies: just add up all the numbers in the `count` variable. It's similarly easy to find the number of cases: just count the rows. And if you want to know the total number of Ahmeds or Sherinas across the years, there's an easy way to do that.

In contrast, it would be more difficult in the Minneapolis election data to find, say, the total number of ballots cast. If you take the seemingly obvious approach and add up the numbers in column I of Table 1.3 (labelled "Total ballots cast"), the result will be *three times* the true number of ballots, because some of the rows contain summaries, not cases.

Indeed, if you wanted to do calculations based on the Minneapolis election data, you would be far better off to put it in a tidy form, like Table 1.4.

ward	precinct	registered	voters	absentee	total.turnout
1	1	28	492	27	0.27
1	4	29	768	26	0.37
1	7	47	291	8	0.16
2	1	63	1011	39	0.36
2	4	53	117	3	0.07
		... and so on for 117 rows			

Table 1.4: The Minneapolis election data in tidy form.

The tidy form is, admittedly, not as attractive as the form published by the Minneapolis government. But the tidy form is much easier to use for the purpose of generating summaries and analyses.

Once data are in a tidy form, you can present them in ways that can be more effective than a formatted spreadsheet. Figure 1.1 presents the turnout in each ward that makes it easy to see how much variation there is within and among precincts.

The tidy format also makes it easier to bring together data from different sources. For instance, to explain the variation in voter turnout, you might want to look at variables such as party affiliation, age, income, etc. Such data might be available on a ward-by-ward basis from other records, such as the (public) voter registration logs and census records. Tidy data can be *wrangle*d into forms that can be

WRANGLE: Data wrangling is the process of reforming, summarizing, and combining data to make it more suitable for a given purpose.

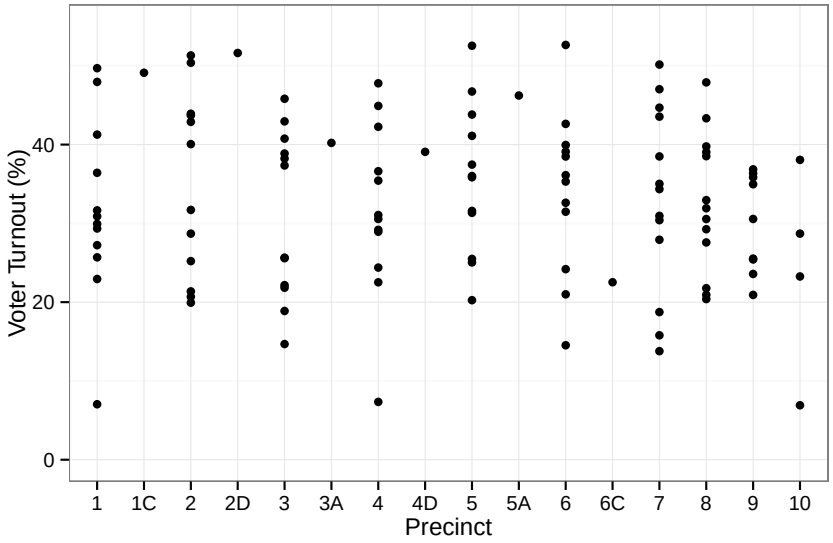


Figure 1.1: A graphical depiction of voter turnout in the different wards.

connected to one another. This would be difficult if you had to deal with an idiosyncratic format for each different source of data.

1.1 *Variables*

In data science, the word *variable* has a different meaning than in mathematics. In algebra, a variable is an unknown quantity. In data, a variable is known; it has been measured. "Variable" refers to a specific quantity or quality that can vary from case to case.

VARIABLE: A quantity or quality that varies from one case to another.

There are two major types of variables: *categorical* and *quantitative* A quantitative variable is just what it sounds like: a number.

A categorical variable tells which category or group a case falls into. For instance, in the baby names data table, `sex` is a categorical variable with two *levels* F and M, standing for female and male. Similarly the `name` variable is categorical. It happens that there are 92,600 different levels for `name`, ranging from Aaron, Ab, and Abbie to Zyhaire, Zylis, and Zymya.

CATEGORICAL: One of the two major kinds of variables. Categorical variables record type or category and often take the form of a word.

QUANTITATIVE: The other of the two major types of variables. A quantitative variable records a numerical attribute.

LEVELS: The individual possibilities for a categorical variable.

1.2 *Cases and what they represent*

As already stated, a row of a tidy data table refers to a case. To this point, you may have little reason to prefer the word *case* to *row*.

When working with a data table, it's important to keep in mind what a case stands for in the real world. Sometimes the meaning is obvious. For instance, Table 1.5 is a tidy data table showing the ballots in the Minneapolis mayoral election in 2013. Each case is an individual voter's ballot. (The voters were directed to mark their

ballot with their first choice, second choice and third choice among the candidates. This is part of a procedure called rank choice voting: `http://vote.minneapolismn.gov/rcv/`.)

Precinct	First	Second	Third	Ward
P-04	undervote	undervote	undervote	W-6
P-06	BOB FINE	MARK ANDREW	undervote	W-10
P-02D	NEAL BAXTER	BETSY HODGES	DON SAMUELS	W-7
P-01	DON SAMUELS	undervote	undervote	W-5
P-03	CAM WINTON	DON SAMUELS	OLE SAVIOR	W-1
... and so on for 80,101 rows				

Table 1.5: Individual ballots in the Minneapolis election. Each voter votes in one ward in one precinct. The ballot marks the voter's first three choices for mayor.

The case in Table 1.5 is a different sort of thing than the case in Table 1.4. In Table 1.4, a case is a ward in a precinct. But in Table 1.5, the case is an individual ballot. Similarly, in the baby names data (Table 1.1), a case is a name and sex and year while in Table 1.2 the case is a name and sex.

When thinking about cases, ask this question: What description would make every case unique? In the vote summary data, a precinct does not uniquely identify a case. Each individual precinct appears in several rows. But each precinct & ward combination appears once and only once. Similarly, in Table 1.1, name & sex do not specify a unique case. Rather, you need the combination of name & sex & year to identify a unique row.

Example: Runners and Races Table 1.6 shows some of the results from a 10-mile running race held each year in Washington, D.C.

name.yob	sex	age	year	gun
jane polanek 1974	F	32	2006	114.50
jane poole 1948	F	55	2003	92.72
jane poole 1948	F	56	2004	87.28
jane poole 1948	F	57	2005	85.05
jane poole 1948	F	58	2006	80.75
jane poole 1948	F	59	2007	78.53
jane schultz 1964	F	35	1999	91.37
jane schultz 1964	F	37	2001	79.13
jane schultz 1964	F	38	2002	76.83
jane schultz 1964	F	39	2003	82.70
jane schultz 1964	F	40	2004	87.92
jane schultz 1964	F	41	2005	91.47
jane schultz 1964	F	42	2006	88.43
jane smith 1952	F	47	1999	90.60
jane smith 1952	F	49	2001	97.87
... and so on for 41,248 rows				

Table 1.6: An excerpt of runners' performance over the years in a 10-mile race.

What's the meaning of a case here? It's tempting to think that a case is a person. After all, it's people who run road races. But notice that individuals appear more than once: Jane Poole ran each year from 2003 to 2007. (Her times improved consistently as she got older!) Jane Smith ran in the races from 1999 to 2006, missing only the year 2000 race. This suggests that the case is a runner in one year's race.

CODEBOOK: A document separate from the data table detailing the meaning of each variable in the table, how levels of categorical variables are encoded, units for quantitative variables, etc.

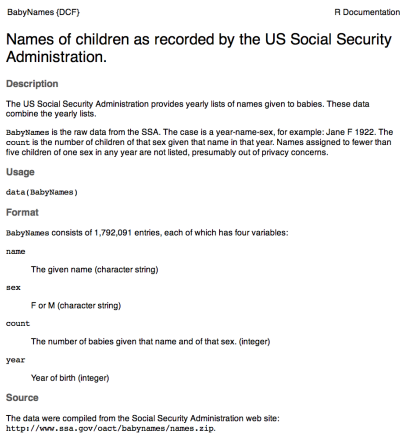
1.3 *The Codebook*

Data tables do not necessarily display all the variables needed to figure out what makes each row unique. For such information, you sometimes need to look at the documentation of how the data were collected and what the variables mean.

The *codebook* is a document — separate from the data table — that describes various aspects of how the data were collected, what the variables mean and what the different levels of categorical variables refer to. Figure 1.2 shows the codebook for the `BabyNames` data in Figure 1.1.

The word "codebook" comes from the days when data was encoded for the computer in ways that make it hard for a human to read. A codebook should also include information about how the data were collected and what constitutes a case.

For the runners data in Table 1.6, the codebook tells you that the meaning of the `gun` variable is the time from when the start gun went off to when the runner crosses the finish line and that the unit of measurement is "minutes." It should also state what might be obvious: that `age` is the person's age in years and `sex` has two levels, male and female, represented by M and F.



BabyNames (DCF) R Documentation

Names of children as recorded by the US Social Security Administration.

Description

The US Social Security Administration provides yearly lists of names given to babies. These data combine the yearly lists.

`BabyNames` is the raw data from the SSA. The case is a year-name-sex, for example: Jane F 1922. The `count` is the number of children of that sex given that name in that year. Names assigned to fewer than five children of one sex in any year are not listed, presumably out of privacy concerns.

Usage

```
data(BabyNames)
```

Format

`BabyNames` consists of 1,792,091 entries, each of which has four variables:

`name`
: The given name (character string)

`sex`
: F or M (character string)

`count`
: The number of babies given that name and of that sex. (integer)

`year`
: Year of birth (integer)

Source

The data were compiled from the Social Security Administration web site: `http://www.ssa.gov/oact/babynames/names.zip`.

Figure 1.2: The codebook for the `BabyNames` data table.

1.4 *Multiple Tables*

It's often the case that creating a meaningful display of data involves combining data from different sources and about different kinds of things. For instance, you might want your analysis of the runners' performance data in Table 1.6 to include temperature and precipitation data for each year's race. Such weather data is likely contained in a table of daily weather measurements.

In many circumstances, there will be multiple tidy tables, each of which contains information relative to your analysis but has a different kind of thing as a case. Chapter 10 is about techniques combining data from such different tables. For now, keep in mind that being tidy is not about shoving everything into one table.

Exercises

Problem 1.1
Here is an excerpt from the baby-name data table in `"DataComputing::BabyNames"`.

name	sex	count	year
Taffy	F	19	1970
Liliana	F	162	1973
Stan	M	55	1975
Nettie	F	45	1978
Kateria	F	8	1980
... and so on for 1,792,091 rows			

Consider these five entities, that appear in the table shown above (a) through (e):

a) `Taffy` b) `year` c) `sex` d) `name` e) `count`

For each, choose one of the following:

1. It's a categorical variable.
2. It's a quantitative variable.
3. It's the value of a variable for a particular case.

Problem 1.2
What's not tidy about this table?

`president`	`in office`	`number of states`
Lincoln, Abraham	1861-1865	it depends
George Washington	1791-1799	16
Martin Van Buren	1837 to 1841	26

Table 1.7: An untidy table

Problem 1.3
Re-write Table 1.7 in a tidy form. Take care to render the information about years and about the number of states as numbers.

Problem 1.4
Here are three different organizations (**A**, **B**, and **C**) of the same data:

Data Table A

Year	Algeria	Brazil	Columbia
2000	7	12	16
2001	9	14	18

Data Table B

Country	Y2000	Y2001
Algeria	7	9
Brazil	12	14
Columbia	16	18

Data Table C

Country	Year	Value
Algeria	2000	7
Algeria	2001	9
Brazil	2000	12
Columbia	2001	18
Columbia	2000	16
Brazil	2001	14

1. What are the variables in each table?
2. What is the meaning of a case for each table? Here are some possible choices.
   - A country
   - A country in a year
   - A year

Problem 1.5
The codebook for several data tables relating to airports, airlines, and airline flights in the US is published at `https://cran.r-project.org/web/packages/nycflights13/nycflights13.pdf`.

Within that document is the codebook for the data table `airports`.

1. How many variables are there?
2. What do the cases represent?
3. For each variable, make a reasonable guess about whether the values will be numerical or quantitative.