5
Introduction to Data Graphics

Data graphics provide one of the most accessible, compelling, and expressive modes to investigate and depict patterns in data. This chapter presents examples of standard kinds of data graphics: what they are used for and how to read them. To start, you'll make simple examples of graphics using an interactive tool. Later, in Chapter 6, you'll see a unifying framework — a grammar — for describing and specifying graphics, so that you can create custom graphics types that support displaying data in a purposeful way.

5.1 Common Kinds of Graphs

There are many different genres of data graphics, and many different variations on each genre. Here are some commonly encountered kinds.

- **Scatterplots** showing relationships between two or more variables.
- **Displays of distribution**, such as histograms.
- **Bar charts**, comparing values of a single variable across groups.
- **Maps**, showing how a variable relates to geography.
- **Network diagrams**, showing how entities are connected to one another.

Scatter plots

The main purpose of a scatter plot is to show the relationship between two variables across several or many cases. Most often, there is a Cartesian coordinate system in which the x-axis represents one variable and the y-axis the value of a second variable.

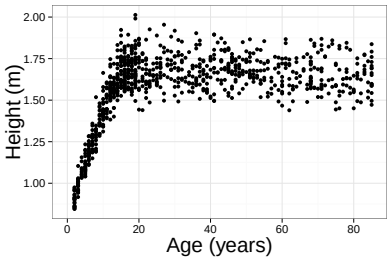


Figure 5.1: A scatter plot.

EXAMPLE: GROWING UP. The NCHS data table gives medical and morphometric measurements of individual people. The scatter plot in Figure 5.1 shows the relationship between two of the variables, height and age. Each dot is one case. The position of that dot signifies the value of the two variables for that case.

Scatterplots are useful for visualizing a simple relationship between two variables. For instance, you can see in the 5.1 the familiar pattern of growth in height from birth to the late teens.

Displays of Distribution

A histogram shows how many cases fall into given ranges of the variable. For instance, Figure 5.2 is a histogram of heights from NCHS. The most common height is about 1.65 m — that's the location of the tallest bar. Only a handful are taller than 2.0 m.

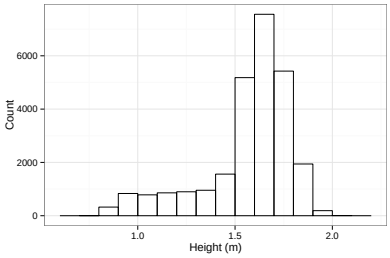


Figure 5.2: A histogram.

A simple alternative to a histogram is a *frequency polygon*. Frequency polygons let you break things up by other variables. Figure 5.3 shows the distribution of height for each sex, separately.

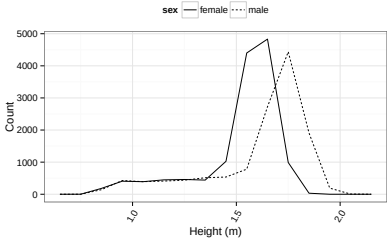


Figure 5.3: A frequency polygon

Bar Charts

The familiar bar chart is effective when the objective is to compare a few different quantities.

EXAMPLE: SMOKING AND DEATH. Based on the NCHS data, how likely is a person to have died during the follow-up period, based on their age and whether they smoke? It's easy to compare bars to their neighbors. From Figure 5.4, for instance, you can see that at each age, non-smokers were more likely to survive.

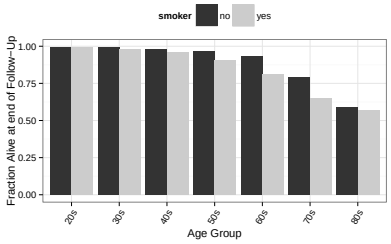


Figure 5.4: A bar chart

Maps

Using a map to display data geographically helps both to identify particular cases and to show spatial patterns and discrepancy. The map in Figure 5.5 shows oil production in each country. That is, the shading of each country represents the variable `oilProd` from `DataComputing::CountryData`. This sort of map, where the fill color of each region reflects the value of a variable, is sometimes called a *choropleth map*.

oil-map-1.pdf

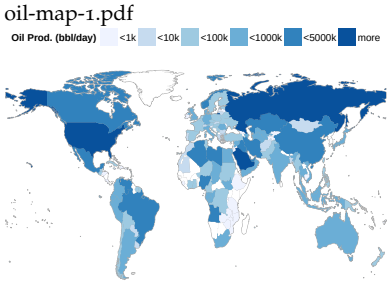


Figure 5.5: A choropleth map

Networks

A *network* is a set of connections, called *edges*, between elements, called *vertices*. A vertex corresponds to a case. The network describes which vertices are connected to other vertices.

The `DataComputing::NCI60` data set is about the genetics of cancer. The data set looks at more than 40,000 probes for the expression of genes, in each of 60 cancers. In the network here, a vertex is a given cell line. Every vertex is depicted as a dot. The dot's color and label gives the type of cancer involved. These are Ovarian, Colon, Central Nervous System, Melanoma, Renal, Breast, and Lung cancers. The edges between vertices show pairs of cell lines that had a strong correlation in gene expression.

The network shows that the melanoma cell lines (ME) are closely related to each other and not so much to other cell lines. The same is true for colon cancer cell lines (CO) and for central nervous system (CN) cell lines.

cancer-network-1.pdf

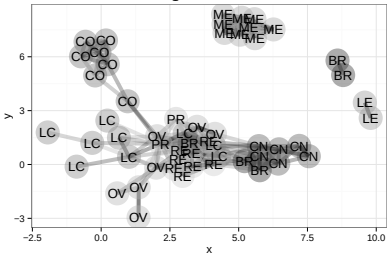


Figure 5.6: A network diagram

5.2 *Constructing Graphics Interactively*

There is a simple pattern to creating a data graphic:

1. Choose or create the glyph-ready data table that will be graphed.
2. Select the kind of graphic: scatterplot, bar chart, map, etc.
3. Decide which variables from the data table will be assigned to which roles in the graphic: *x*- and *y*-coordinates, bar lengths, colors, sizes, etc. This is called *mapping* a variable to a graphical attribute.

MAPPING: specifying which particular graphical attribute is to represent a variable. The word "mapping" is drawn from mathematics, an association between two sets of items, and has nothing to do with geographical maps.

Chapter 8 introduces R commands such as `ggplot()` for drawing graphics. In this chapter, you will use interactive programs to generate the graphics and the corresponding R commands. The commands can be pasted into an R chunk in an `.Rmd` file.[1]

[1] Never put the interactive commands into an `.Rmd` file. There is no one to interact with the session created when compiling. Instead, use the interactive commands in the console to generate the graphics commands, and paste the commands into an R chunk. Instructions for installing `DataComputing` and other packages are posted at `http://Data-Computing.org/` under "Software and Data."

The `DataComputing` package provides several interactive graphing functions. They are interactive in allowing you to use the mouse to specify which variables map to which graphical attributes. The functions are:

- `scatterGraphHelper()`
- `barGraphHelper()`
- `distributionGraphHelper()`
- `WorldMap()`
- `USMap()`

The first argument to each of these functions is a data table whose cases you want to display graphically. The map-making functions also require two more arguments: `key=` specifies the variable in the data table that identifies the country or state for each case. `fill=` specifies the variable to be used for shading each country.

Drawing networks will be introduced later.

Scatter Plots

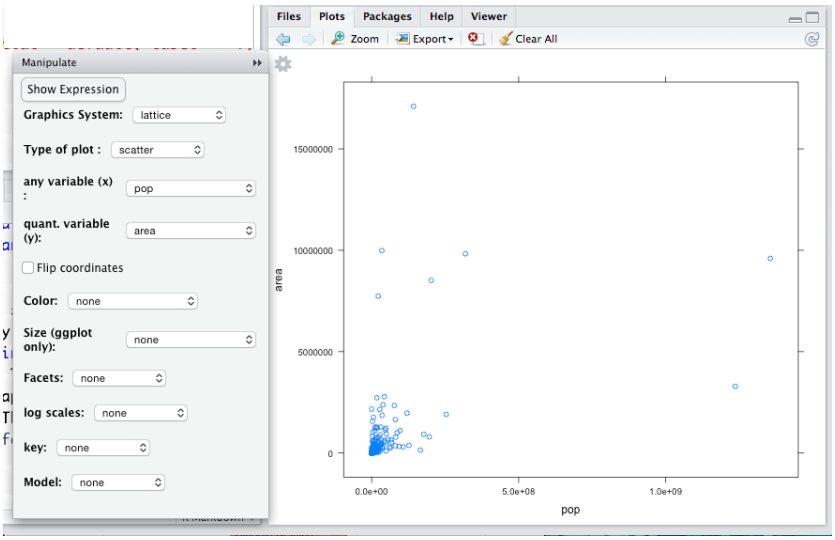


Figure 5.7: The interactive plot tab created using `scatterGraphHelper(CountryData)`. By default, the first two quantitative variables in a data table are mapped to the x- and y-axes. Change this using the drop-down menu to make the scatter plot of interest to you.

Consider the relationship between birth rate and death rate among the countries in `CountryData`. The variables `birth` and `death` that give these rates (in births per 1000 people per year).

An appropriate graphic modality is a scatter plot: birth rate against death rate. To make the graph, use the software appropriate for this modality, namely `scatterGraphHelper()`. As an argument, give the data table from which the variables are taken: `CountryData`.

```
scatterGraphHelper(CountryData)
```

That one simple command gives two essential details: what data table to use and what modality of graph to make. After giving that command, something like Figure 5.7 should appear in your "Plots" tab.

Notice three components of the plots tab:

1. A coordinate grid with dots.
2. A menu for mapping variables to attributes.
3. A small gear icon: 

If you don't see the menu on your system, click on the gear icon. If the menu runs off the bottom of the screen, make the "Plots" tab taller.

By default, the first two quantitative variables, `area` and `pop`, are being used to define the frame. This is not usually what you want.

You can use the `scatterGraphHelper()` menu (Figure 5.8) set the frame to be `death` *versus* `birth`. The overall pattern is U-shaped; both low and high birth rates are associated with high death rates, while birth rates in the middle tend to have lower death rates.

VERSUS: One against the other. The convention is y versus x.

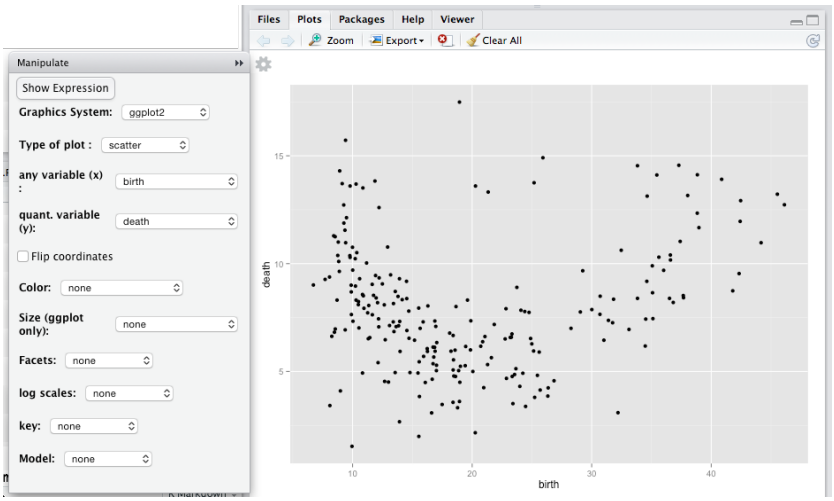
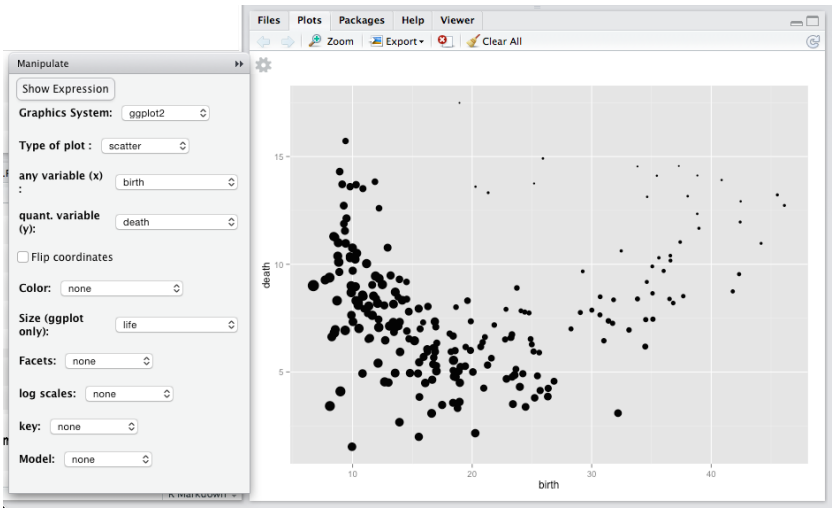


Figure 5.8: Using the `scatterGraphHelper()` menu to set the variables displayed on the x- and y-axes to be birth and rates.

The glyphs in scatterplots — dots here — can have graphical attributes besides from their position in the frame. Standard ones include fill color, shape, size, transparency, and border color. In Figure 5.9, life expectancy is mapped to size. You can see that, for countries with high life expectancy, the death rate is high when the birth rate is low. For countries with low life expectancies, the death rate is high when the birth rate is low.



That's because when life expectancy is high and birth rate is low, the population tends to be older. Older populations have higher death rates.

Figure 5.9: Graphical attributes such as size, shape, and color can be used to represent additional variables. Here, dot size reflects a country's life expectancy.

Distributions

Frequency polygons or histograms are appropriate for showing how the different values are distributed.

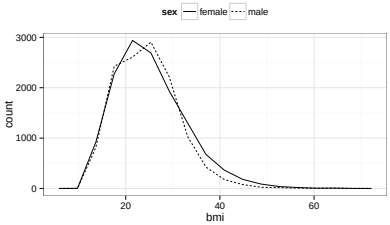


Figure 5.10: The distribution of body mass index shown with a frequency polygon separately for each sex.

```
DistributionGraphHelper(NCHS, format = "frequency polygon")
```

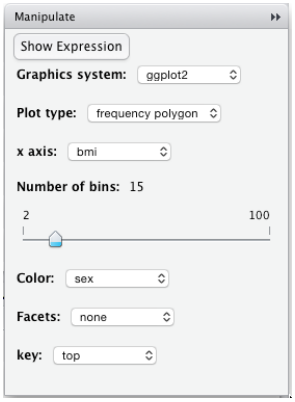


Figure 5.11: Setting the variable mappings for the frequency polygon plot in Figure 5.10 using the `scatterGraphHelper()` menu.

Consider, for instance, how body mass index varies across the subjects in the `NCHS` data. For a distribution, the measured quantity is mapped to the x-axis. The y-values are set by the number of cases at the corresponding x-value. In the `scatterGraphHelper()` menu shown in Figure 5.11, `sex` has been mapped to line type, producing the chart in Figure 5.10.

Bar Plots

Bar charts use a glyph whose *length* reflects the value to be presented. Depending on how variables are mapped to graphical attributes, the plots can tell different aspects of the story.

For instance, the individual ballot choices in the 2013 mayoral election in Minneapolis look like this:

Precinct	First	Second	Third	Ward
P-07	DON SAMUELS	BETSY HODGES	DAN COHEN	W-7
P-05	DON SAMUELS	BETSY HODGES	CHRISTOPHER CLARK	W-9
P-01	DON SAMUELS	BETSY HODGES	MARK ANDREW	W-12
P-03A	MARK ANDREW	undervote	undervote	W-10
P-02	BETSY HODGES	MARK ANDREW	DON SAMUELS	W-9
		... and so on for 80,101 rows		

You might be interested to make a bar chart of the number of first-choice votes that each candidate received. In this case, as is typical, a bit of data wrangling is called for to create glyph-ready data. For now, don't worry about the following commands, which will be introduced later.

```
FirstPlaceTally <-
  Minneapolis2013 %>%
  rename(candidate=First) %>%
  group_by(candidate) %>%
  summarise(total = n())
```

candidate	total
ALICIA K. BENNETT	351
BETSY HODGES	28935
BILL KAHN	97
BOB FINE	2094
CAM WINTON	7511

Table 5.1: First place vote tallys in the `Minneapolis2013` data

There were 37 candidates in the election (!), but most got only a small number of votes. The results can be displayed effectively with a bar chart.

```
barGraphHelper(FirstPlaceTally)
```

The chart shows at a glance that there are just a handful of major candidates. For this plot, the `total` variable was mapped to the y-axis, the candidate was mapped to the x-axis, and the candidates were ordered from lowest to highest total of votes. Look closely and you'll see that "undervote" (meaning no candidate was chosen) beat 29 of the candidates.

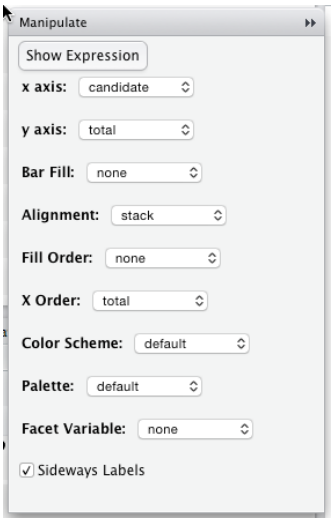


Figure 5.13: The mappings for the vote-tally bar chart in Figure 5.12

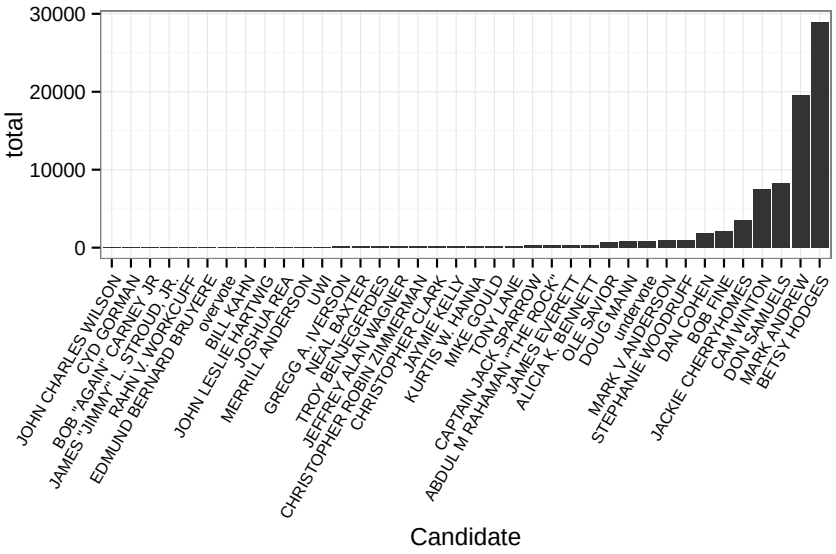


Figure 5.12: A bar chart showing the number of first-place votes given to each candidate.

Making Maps

Showing a variable in a geographical map requires two data tables:

1. A *shape file* giving latitude and longitude of points on the boundaries.

2. The data table for the variable that is to be plotted.

There are shape files for all sorts of geographic entities: countries, states, counties, precincts, and so on. To simplify things, the `DataComputing` package provides two functions: `WorldMap()` with country boundaries and `USMap()` with state boundaries in the US. The shape file is pre-set for these functions; you need only provide a data table with the name of countries (or states) and the variable to be plotted. Figure 5.14 shows how fertility varies from country to country.

```
CountryData %>%
  WorldMap(key="country",fill="fert")
```

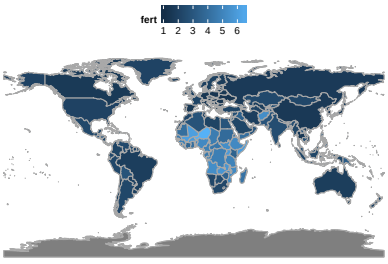
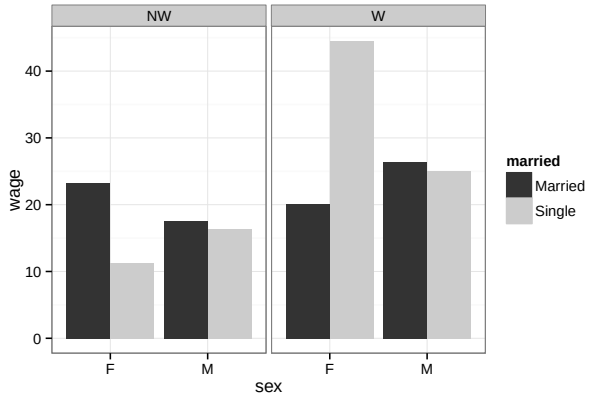


Figure 5.14: A choropleth map of fertility (children born per woman)

5.3 *Exercises*

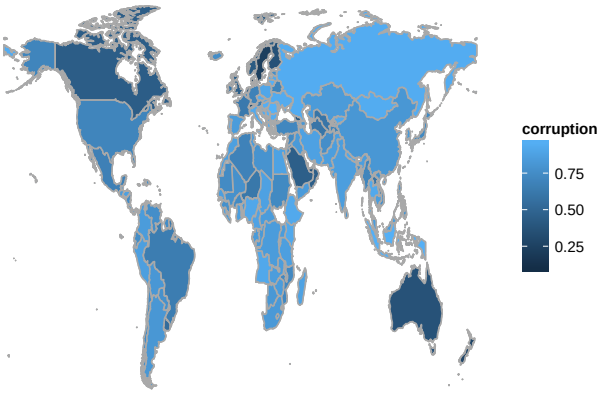
Problem 5.1
Consider this bar graph of the `CPS85` data in the `mosaicData` package:



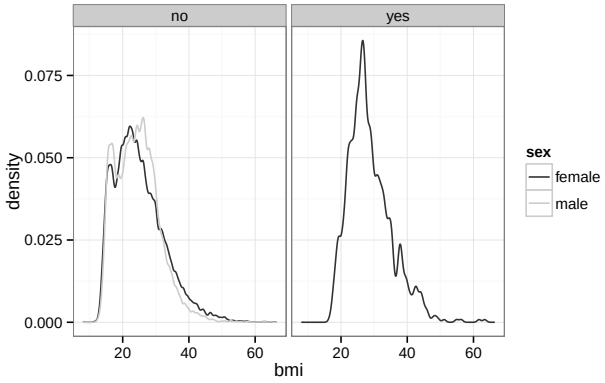
Use `barGraphHelper()` to reconstruct the graph. Start with these commands:

```
library(mosaicData)
library(DataComputing)
barGraphHelper(CPS85)
```

Problem 5.2
Make this map using data from `HappinessIndex` in the `DataComputing` package:



Problem 5.3
Make this graph from the `NCHS` data in the `DataComputing` package.



The "yes" and "no" in the gray bars refer to whether or not the person is pregnant.

Problem 5.4
Using the `CPS85` data table (from the `mosaicData` package) make this graphic: