5. Drawing graphs

Above all else show the data.
–Edward Tufte[1]

Visualising data is one of the most important tasks facing the data analyst. It's important for two distinct but closely related reasons. Firstly, there's the matter of drawing "presentation graphics", displaying your data in a clean, visually appealing fashion makes it easier for your reader to understand what you're trying to tell them. Equally important, perhaps even more important, is the fact that drawing graphs helps *you* to understand the data. To that end, it's important to draw "exploratory graphics" that help you learn about the data as you go about analysing it. These points might seem pretty obvious but I cannot count the number of times I've seen people forget them.

To give a sense of the importance of this chapter, I want to start with a classic illustration of just how powerful a good graph can be. To that end, Figure 5.1 shows a redrawing of one of the most famous data visualisations of all time. This is John Snow's 1854 map of cholera deaths. The map is elegant in its simplicity. In the background we have a street map which helps orient the viewer. Over the top we see a large number of small dots, each one representing the location of a cholera case. The larger symbols show the location of water pumps, labelled by name. Even the most casual inspection of the graph makes it very clear that the source of the outbreak is almost certainly the Broad Street pump. Upon viewing this graph Dr Snow arranged to have the handle removed from the pump and ended the outbreak that had killed over 500 people. Such is the power of a good data visualisation.

The goals in this chapter are twofold. First, to discuss several fairly standard graphs that we use a lot when analysing and presenting data, and second to show you how to create these graphs in JASP. The graphs themselves tend to be pretty straightforward, so in one respect this chapter is pretty simple. Where people usually struggle is learning how to produce graphs, and especially learning how to produce good graphs. Fortunately, learning how to draw graphs in JASP is reasonably simple as long as you're not too picky about what your graph looks like. What I mean when I say this is that JASP has *very* good default graphs, or plots, that most of the time produce a clean, high-quality graphic. However, on those occasions when you do want to do something non-standard, or if you need to make highly specific changes to the figure, then the graphics functionality in JASP is not yet capable of supporting advanced work or detail editing.

[1]The origin of this quote is Tufte's lovely book *The Visual Display of Quantitative Information*.

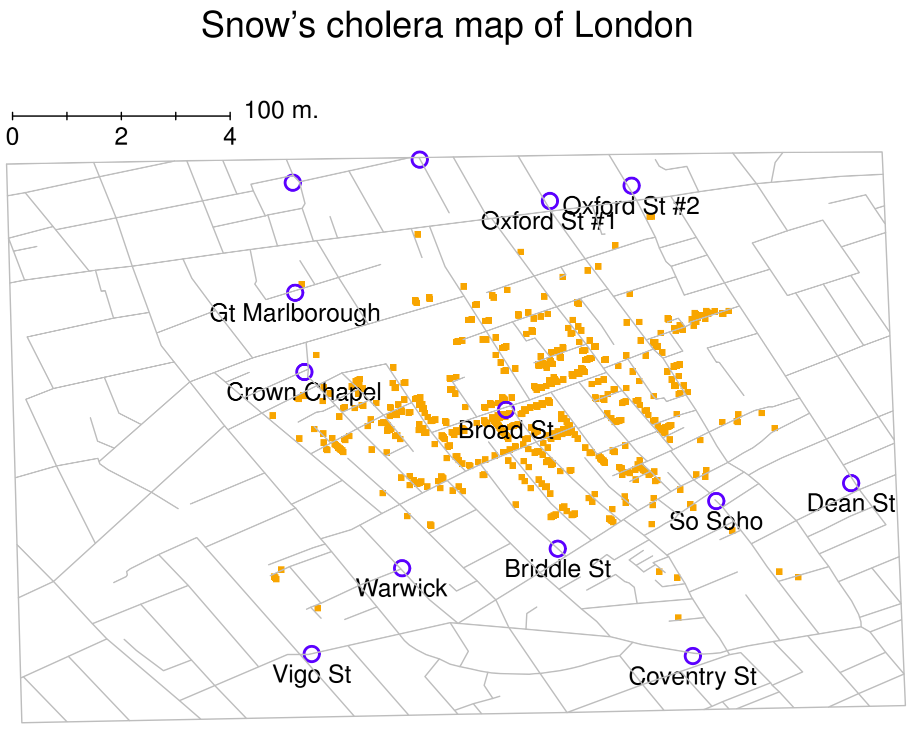
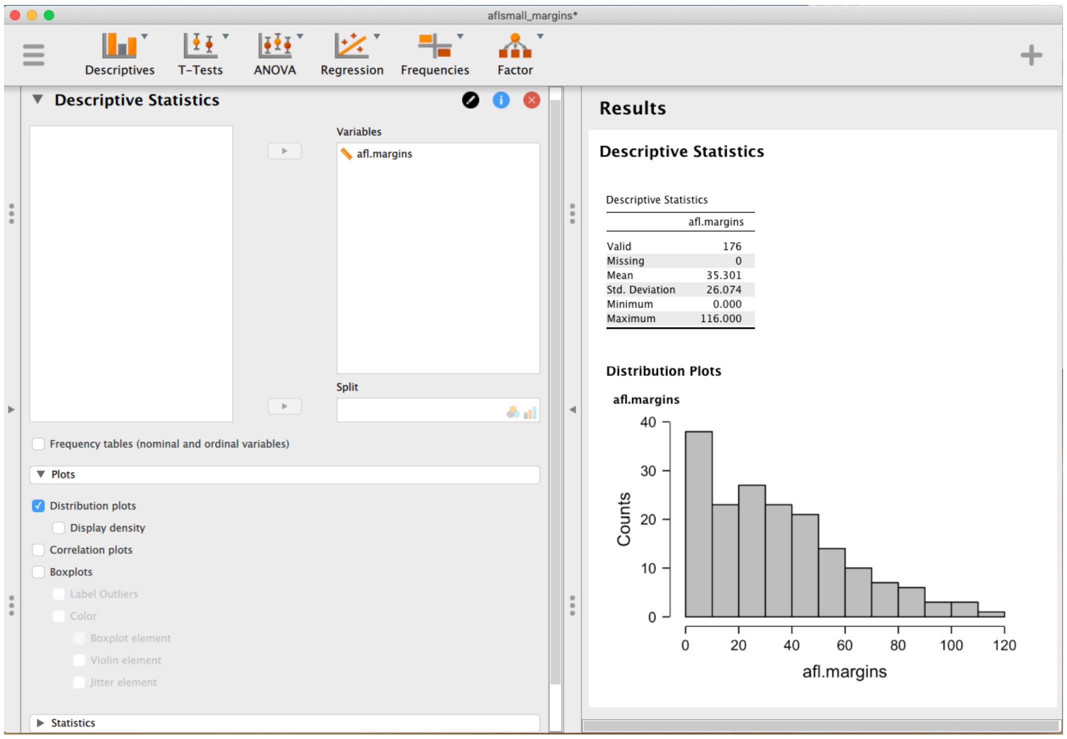


Figure 5.1: A stylised redrawing of John Snow's original cholera map. Each small dot represents the location of a cholera case and each large circle shows the location of a well. As the plot makes clear, the cholera outbreak is centred very closely on the Broad St pump.

5.1 Histograms

Let's begin with the humble **histogram**. Histograms are one of the simplest and most useful ways of visualising data. They make most sense when you have an interval or ratio scale variable (e.g., the `afl.margins` data from Chapter 4) and what you want to do is get an overall impression of the variable. Most of you probably know how histograms work, since they're so widely used, but for the sake of completeness I'll describe them. All you do is divide up the possible values into **bins** and then count the number of observations that fall within each bin. This count is referred to as the frequency or density of the bin and is displayed as a vertical bar. Ihe AFL winning margins data there are 33 games in which the winning margin was less than 10 points and it is this fact that is represented by the height of the leftmost bar that we showed earlier in Chapter 4, Figure 4.2. The earlier graphs were generated using an advanced plotting package in R which, for now, is beyond the capability of JASP. But JASP gets us close, and drawing this histogram in JASP is pretty straightforward. Open up the 'Plots' menu under 'Descriptives' - 'Descriptive Statistics' and click the 'Distribution plots' check box, as in Figure 5.2. JASP defaults to labelling the y-axis as 'Counts' and the x-axis with the variable name. The **bins** are selected automatically. Note that while counts are displayed, the actual values do not matter too much. Rather, what we are really interested in is our impression of the shape of the distribution: is it normally distributed or is there a skew or kurtosis? Our first impressions of these characteristics come from drawing a **histogram**.



Frequency tables (nominal and ordinal variables)

[✓ Distribution plots]
Display density
Correlation plots
Boxplots
Label Outliers
Color
Boxplot element
Violin element
Jitter element

Descriptive Statistics

Descriptive Statistics

	afl.margins
Valid	176
Missing	0
Mean	35.301
Std. Deviation	26.074
Minimum	0.000
Maximum	116.000

Figure 5.2: JASP screenshot showing the 'Distribution plots' option and accompanying histogram.

. . . .

One additional feature that JASP provides is the ability to plot a 'Density' curve on top of this histogram. You can do this by clicking the 'Display density' check box under the 'Plots' options, and this gives us the plot shown in Figure 5.3. A density plot visualises the distribution of data over a continuous interval or time period. This chart is a variation of a histogram that uses **kernel smoothing** to plot values, allowing for smoother distributions by smoothing out the noise. The peaks of a density plot help display where values are concentrated over the interval. An advantage density plots have over histograms is that they are better at determining the distribution shape because they're not affected by the number of bins used (each bar used in a typical histogram). A histogram comprising of only

4 bins wouldn't produce a distinguishable enough shape of distribution as a 20-bin histogram would. However, with density plots, this isn't an issue.

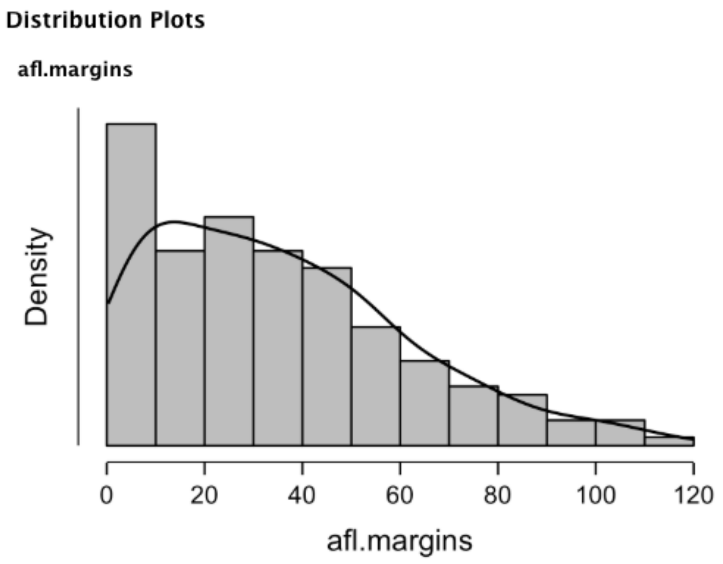


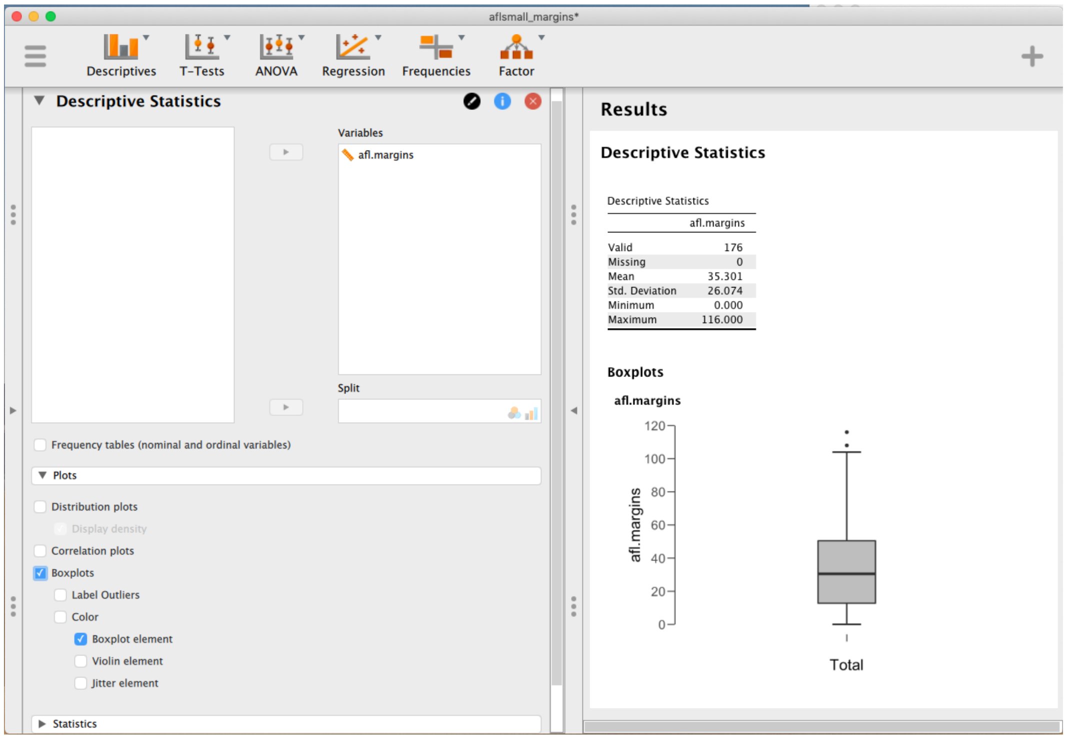
Figure 5.3: A density plot of the `afl.margins` variable plotted in JASP

. . . .

Although this image would need a lot of cleaning up in order to make a good presentation graphic (i.e., one you'd include in a report), it nevertheless does a pretty good job of describing the data. In fact, the big strength of a histogram or density plot is that (properly used) it does show the entire spread of the data, so you can get a pretty good sense about what it looks like. The downside to histograms is that they aren't very compact. Unlike some of the other plots I'll talk about it's hard to cram 20-30 histograms into a single image without overwhelming the viewer. And of course, if your data are nominal scale then histograms are useless.

5.2 Boxplots

Another alternative to histograms is a **boxplot**, sometimes called a "box and whiskers" plot. Like histograms they're most suited to interval or ratio scale data. The idea behind a boxplot is to provide a simple visual depiction of the median, the interquartile range, and the range of the data. And because they do so in a fairly compact way boxplots have become a very popular statistical graphic, especially during the exploratory stage of data analysis when you're trying to understand the data yourself. Let's have a look at how they work, again using the `afl.margins` data as our example.



Descriptive Statistics

Variables: afl.margins

- [ ] Frequency tables (nominal and ordinal variables)

Plots

- [ ] Distribution plots
  - [ ] Display density
- [ ] Correlation plots
- [✓] Boxplots
  - [ ] Label Outliers
  - [ ] Color
    - [✓] Boxplot element
    - [ ] Violin element
    - [ ] Jitter element

Statistics

Results

Descriptive Statistics

Descriptive Statistics

	afl.margins
Valid	176
Missing	0
Mean	35.301
Std. Deviation	26.074
Minimum	0.000
Maximum	116.000

Boxplots

afl.margins

Figure 5.4: A box plot of the `afl.margins` variable plotted in JASP

. . . .

The easiest way to describe what a boxplot looks like is just to draw one. Click on the 'Boxplots' check box and you will get the plot shown on the lower right of Figure 5.4. By default, JASP has drawn the most basic boxplot possible. When you look at this plot this is how you should interpret it: the thick line in the middle of the box is the median; the box itself spans the range from the 25th percentile to the 75th percentile; and the "whiskers" go out to the most extreme data point that doesn't exceed a certain bound. By default, this value is 1.5 times the interquartile range (IQR), calculated as `25th percentile - (1.5*IQR)` for the lower boundary, and `75th percentile + (1.5*IQR)` for the upper boundary. Any observation whose value falls outside this range is plotted as a circle or dot instead of being covered by the whiskers, and is commonly referred to as an **outlier**. For our AFL margins data there are two observations that fall outside this range, and these observations are plotted as dots (the upper boundary is 107, and looking over the data column in the spreadsheet there are two observations with values higher than this, 108 and 116, so these are the dots).

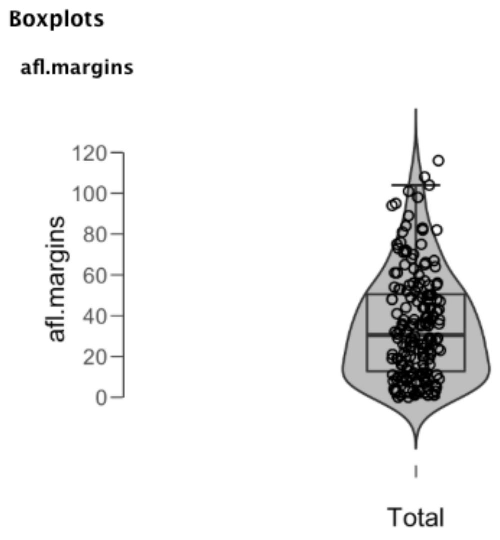


Figure 5.5: A violin plot of the `afl.margins` variable plotted in JASP, also showing a box plot and data points

. . . .

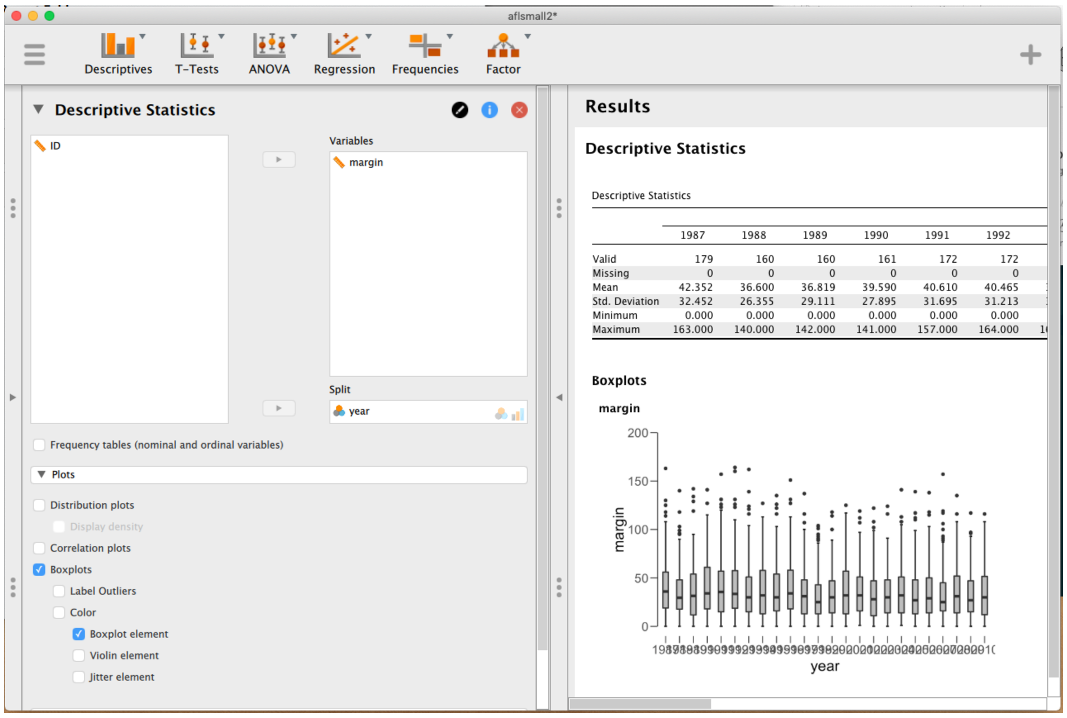
5.2.1 Violin plots

A variation to the traditional box plot is the violin plot. Violin plots are similar to box plots except that they also show the kernel probability density of the data at different values. Typically, violin plots will include a marker for the median of the data and a box indicating the interquartile range, as in standard box plots. In JASP you can achieve this sort of functionality by checking both the 'Violin element' and the 'Boxplot element' check boxes. See Figure 5.5, which also has the data plotted (select the 'Jitter element' check box to show the actual data points on the plot).

5.2.2 Drawing multiple boxplots

One last thing. What if you want to draw multiple boxplots at once? Suppose, for instance, I wanted separate boxplots showing the AFL margins not just for 2010 but for every year between 1987 and 2010. To do that the first thing we'll have to do is find the data. These are stored in the `aflsmall2.csv` file. So let's load it into JASP and see what is in it. You will see that it is a pretty big data set. It contains 4296 games and the variables that we're interested in. What we want to do is have JASP draw boxplots for the `margin` variable, but plotted separately for each `year`. The way to do this is to first change `year` to a nominal variable, then `year` across into the 'Split' box.

The result is shown in Figure 5.6. This version of the box plot, split by year, gives a sense of why it's sometimes useful to choose box plots instead of histograms. It's possible to get a good sense of what the data look like from year to year without getting overwhelmed with too much detail. Now imagine what would have happened if I'd tried to cram 24 histograms into this space: no chance at all that the reader is going to learn anything useful.



Descriptive Statistics

Descriptive Statistics

	1987	1988	1989	1990	1991	1992
Valid	179	160	160	161	172	172
Missing	0	0	0	0	0	0
Mean	42.352	36.600	36.819	39.590	40.610	40.465
Std. Deviation	32.452	26.355	29.111	27.895	31.695	31.213
Minimum	0.000	0.000	0.000	0.000	0.000	0.000
Maximum	163.000	140.000	142.000	141.000	157.000	164.000

Figure 5.6: Multiple boxplots plotted in JASP, for the `margin` by `year` variables in the `aflsmall2` data set

. . . .

5.3 Saving image files using JASP

Hold on, you might be thinking. What's the good of being able to draw pretty pictures in JASP if I can't save them and send them to friends to brag about how awesome my data is? How do I save the picture? Simples. Just click on the triangle beside the header of your plot and select 'Save Image As'. You can save it in one of several formats, including 'png', 'pdf', 'eps', or 'tif'. These formats all produce nice images that you can the send to your friends, or (perhaps more importantly) include in your assignments or papers.

5.4 Summary

Perhaps I'm a simple minded person, but I love pictures. Every time I write a new scientific paper one of the first things I do is sit down and think about what the pictures will be. In my head an article is really just a sequence of pictures linked together by a story. All the rest of it is just window dressing. What I'm really trying to say here is that the human visual system is a very powerful data analysis tool. Give it the right kind of information and it will supply a human reader with a massive amount of knowledge very quickly. Not for nothing do we have the saying "a picture is worth a thousand words". With that in mind, I think that this is one of the most important chapters in the book. The topics covered were:

- *Common plots*. Much of the chapter was focused on some of the standard graphs that statisticians like to produce: histograms (Section 5.1) and boxplots (Section 5.2)
- *Saving image files*. Importantly, we also covered how to export your pictures (Section 5.3)

One final thing to point out. While JASP produces some really neat default graphics, editing the plots is currently not possible. For more advanced graphics and plotting capability the packages available in R are much more powerful. One of the most popular graphics systems is provided by the `ggplot2` package (see `http://ggplot2.org/`), which is loosely based on "The grammar of graphics" (Wilkinson, Wills, Rope, Norton, and Dubbs 2006). It's not for novices. You need to have a pretty good grasp of R before you can start using it, and even then it takes a while to really get the hang of it. But when you're ready it's worth taking the time to teach yourself, because it's a much more powerful and cleaner system.