DATA VISUALISATION

LAB WORKSHEET 9

Creator: Dr Mark Taylor

What's Happening in this Document?

We are going to go through the tiresome experience of downloading a real dataset from the web, where you might know what you want to do with it, but it's not in a format that's well-suited to what you want to do. This isn't an exhaustive set of different kinds of problems you might run into: the "R for Data Science Book" (in the Module Outline) walks you through several more. But this should give you a flavour of the amount of data cleaning you might have to do before you can start visualising data.

We'll need to do several things in order to draw the graphs we're interested in:

- First, we'll need to load the data into RStudio. This has been fairly low-stress so far; it isn't always.
- Second, we'll need to manipulate the data in order to get it into the shape we want. This involves several different steps.
- Finally, we'll need to ensure that our loaded, manipulated data looks how we want.

For this worksheet, we're going to use some recent World Bank data on the quality of democracy in different countries.

Q-Step A step-change in quantitative social science skills Funded by the Nuffield Foundation, ESRC and HEFCE
Nuffield Foundation
Sheffield Methods Institute.
The University Of Sheffield.

Amongst other things, the file contains information on how people in different countries feel they have voice and accountability, and how this has changed over time. This data is the original dataset where we got our data for the previous worksheet. Now, we are going to learn how to clean a dataset so that in the end we are able to draw a graph of how voice and accountability has varied in France, Italy, the UK, and USA.

The dataset is included on Blackboard, please download it and place it in your working directory.

Before we begin, please read over the worksheet (before loading it in into R) and have a look at what the dataset looks like in excel in order to give you a better idea of what we are doing.

Getting started

Let's start by loading some packages:

```
library(tidyverse)
library(readxl)
```

and attempting to load the data. We're going to use read_excel(), which is a new command; it's handy for loading Excel sheets (as opposed to csvs, and so on).

Trying to load the data

Let's try to load the data. Please make sure you have downloaded the file from the Q-Step Website (or from this link: https://bit.ly/lab_worksheet_9) and that you have saved it to where you have set your working directory:

```
world_bank_stuff <-
  read_excel("wgidataset_2020.xlsx")
```

We can see from the top right pane that this hasn't really worked. If we open the Excel file, we can see it consists of loads of different sheets, containing information about different variables. As we browse through, we can see that the one we're after is in the second sheet. Let's try loading it.

```
world_bank_stuff <-
  read_excel("wgidataset_2020.xlsx", sheet = 2)
```

While this looks better, R's given us some warning signs, and rightly so. It looks like what's happened is that there's loads of redundant stuff at the top of the file – this is metadata, which would ideally have been stored separately.

But we can't be precious about this. The World Bank have been good enough to give us this data; let's make the most of it.

Something we can do instead is to specify the range of the data we want to use. We can see the data seems to have 229 rows (when we bear in mind that the first row's been read in as variable names); we can also see the first 14 rows seem to be redundant. So we can specify the range of the data so we only include the stuff we want, and exclude the stuff we don't. We can do this like so.

```
world_bank_stuff <-
  read_excel("wgidataset_2020.xlsx",
             sheet = 2,
             range = cell_rows(15:229))
```

Getting there. But this still isn't quite what we want. There's lots of missing data. This shouldn't come as a surprise, as plenty of countries make it difficult to find out about their residents. However, ideally these cells would have just been blank; instead, they're specified with #N/A. So we need to tell R explicitly that that corresponds to missingness. Like so:

```
world_bank_stuff <-
  read_excel("wgidataset_2020.xlsx",
             sheet = 2,
             range = cell_rows(15:229),
             na = "#N/A")
```

Right. We've loaded the data. This is usually more painless than this…

Manipulating the data

So, what do our variables look like?

- We've got a variable for country name. This isn't as clean as it could be; it's annoying to have slashes in our variable names.
- We've got a variable for country code. This is a bit easier to deal with.
- We've then got, for each year since 1996 (it used to be measured every two years), a measure of an estimate, standard error, number of data sources, percentile rank, lower bound, and upper bound, on the variable in question.

The first one's a bit annoying; the last one's a real issue. This is what's referred to as wide data. In the tidyverse, we want every row to be a case, and every column to be a variable; this isn't the case here, where each row contains a lot of different cases (at different time points), and several columns consist of the same variables, just at different time points.

Let's start by just keeping the country names, country codes, and estimates for each year. It's very fiddly to keep everything; let's get the data down to a manageable size.

```
world_bank_stuff %>%
  select(`Country/Territory`, Code, starts_with("Estimate"))
```

This is a start. However, it's not ideal; we'll never remember what each of these estimates corresponds to, which is the year in which the data were collected. So, following what we did in our last worksheet, let's do loads of renaming variables. Please stick this on the end of the previous command:

```
%>%
  rename(country = `Country/Territory`,
         `1996` = Estimate...3,
         `1998` = Estimate...9,
         `2000` = Estimate...15,
         `2002` = Estimate...21,
         `2003` = Estimate...27,
         `2004` = Estimate...33,
         `2005` = Estimate...39,
         `2006` = Estimate...45,
         `2007` = Estimate...51,
         `2008` = Estimate...57,
         `2009` = Estimate...63,
         `2010` = Estimate...69,
         `2011` = Estimate...75,
         `2012` = Estimate...81,
         `2013` = Estimate...87,
         `2014` = Estimate...93,
         `2015` = Estimate...99,
         `2016` = Estimate...105,
         `2017` = Estimate...111,
         `2018` = Estimate...117,
         `2019` = Estimate...123)
```

While this is a bit painful, we're at least getting somewhere. But what we really want is a long dataset, where we have an observation for the relevant value (each case of estimate) for each year for each country, not a single row for each country.

We can address this using the gather command. This reorganises data in the way that we want, converting it from wide to long. (We can use spread to convert from long to wide; we might do this if we had a variable that explained which variable each value corresponded to. This is more common than you might think.)

To do this, we need to tell R:

What variables we're gathering together. In this case, we want all our year variables to end up a single variable, paired up with something that tells us what year we're looking at

What we want to call that new variable full of numbers

What we want to call the variable where the values are the original column names

Let's call them accountability (which is the thing being measured) and year. Like so: please stick this on the end of what we've already got.

```
    %>%
      gather(`1996`:`2019`,
             key = "year",
             value = "accountability")
```

Has this worked OK?

Assuming it has, let's...

Draw a graph

Please stick the following on the end of what we've already got. Having spent a bunch of time putting the thing together, it looks like we can finally draw a fairly simple graph; let's draw a graph of how perceived accountability in the US has changed since 1996. Please stick this on the end of what we've already got.

```
    %>%
      filter(country == "United States")  %>%
      ggplot() +
      aes(x = year, y = accountability) +
      geom_line()
```

This hasn't worked brilliantly. The problem is, despite the year variable consisting of numbers, R's treating it as categorical; that's because these values all used to be variable names. We can address this by tweaking the x variable, which is currently just year, into as.numeric(year), which forces R to treat is as a number. So please tweak the last change you made so it looks like the following:

```
      ggplot() +
      aes(x = as.numeric(year), y = accountability) +
      geom_line()
```

Finally, let's tweak the graph so that we can do what we originally set out to do. Let's compare a few countries. Let's change our old filter() command to the following:

```
    filter(country == "United States" |
               country == "United Kingdom" |
               country == "France" |
               country == "Italy")
```

and add some faceting:

```
    +
      facet_wrap(~ country)
```

We’ve now drawn the graph we originally set out to draw. While this process can be pretty tortured, this is what it’s actually like for real; we spend a lot of time getting our data into the format we want.

Going further

Were we right to discard loads of data earlier? It seems like there's a lot of uncertainty associated with some of these measures, and that this is particularly an issue for parts of the world where accountability is poorly-measured (for example, there's only one indicator, rather than several). So maybe we want to include those upper and lower bounds, as well as overall rank estimate. How can we do that? Can we gather in such a way that we're generating three new variables, rather than just one?

One way to do it is to generate three new objects, rather than just one, and merge them together. As we're making them out of a single original dataset, we can be confident that the names used are consistent.

So, let's do that. Let's repeat what we did before, but for three different variables rather than just one, and see what happens:

First, lower bounds:

```
lower <-
  world_bank_stuff %>%
  select(`Country/Territory`, Code, starts_with("Lower")) %>%
  rename(country = `Country/Territory`,
         `1996` = Lower...7,
         `1998` = Lower...13,
         `2000` = Lower...19,
         `2002` = Lower...25,
         `2003` = Lower...31,
         `2004` = Lower...37,
         `2005` = Lower...43,
         `2006` = Lower...49,
         `2007` = Lower...55,
         `2008` = Lower...61,
         `2009` = Lower...67,
         `2010` = Lower...73,
         `2011` = Lower...79,
         `2012` = Lower...85,
         `2013` = Lower...91,
         `2014` = Lower...97,
         `2015` = Lower...103,
         `2016` = Lower...109,
         `2017` = Lower...115,
         `2018` = Lower...121,
         `2019` = Lower...127 ) %>%
  gather(`1996`:`2019`,
         key = "year",
         value = "Lower")
```

Next, upper bounds,

```r
upper <-
  world_bank_stuff %>%
  select(`Country/Territory`, Code, starts_with("Upper")) %>%
  rename(country = `Country/Territory`,
         `1996` = Upper...8,
         `1998` = Upper...14,
         `2000` = Upper...20,
         `2002` = Upper...26,
         `2003` = Upper...32,
         `2004` = Upper...38,
         `2005` = Upper...44,
         `2006` = Upper...50,
         `2007` = Upper...56,
         `2008` = Upper...62,
         `2009` = Upper...68,
         `2010` = Upper...74,
         `2011` = Upper...80,
         `2012` = Upper...86,
         `2013` = Upper...92,
         `2014` = Upper...98,
         `2015` = Upper...104,
         `2016` = Upper...110,
         `2017` = Upper...116,
         `2018` = Upper...122,
         `2019` = Upper...128) %>%
  gather(`1996`:`2019`,
         key = "year",
         value = "Upper")
```

Finally, rank estimates.

```
rank_estimate <-
  world_bank_stuff %>%
  select(`Country/Territory`, Code, starts_with("Rank")) %>%
  rename(country = `Country/Territory`,
`1996` = Rank...6,
         `1998` = Rank...12,
         `2000` = Rank...18,
         `2002` = Rank...24,
         `2003` = Rank...30,
         `2004` = Rank...36,
         `2005` = Rank...42,
         `2006` = Rank...48,
         `2007` = Rank...54,
         `2008` = Rank...60,
         `2009` = Rank...66,
         `2010` = Rank...72,
         `2011` = Rank...78,
         `2012` = Rank...84,
         `2013` = Rank...90,
         `2014` = Rank...96,
         `2015` = Rank...102,
         `2016` = Rank...108,
         `2017` = Rank...114,
         `2018` = Rank...120,
         `2019` = Rank...126) %>%
  gather(`1996`:`2019`,
         key = "year",
         value = "Rank estimate")
```

Then we can merge lower and rank:

```
lower_and_rank <-
  merge(lower,
        rank_estimate,
        id = c("country", "Code"))
```

And then we can merge all three:

```
all_three <-
  merge(lower_and_rank,
        upper,
        id = c("country", "Code"))
```

and, finally, gather once again.

```
for_final_plot <-
  all_three %>%
  gather(Lower, `Rank estimate`, Upper, key = "measure", value = "combined")
```

Finally we're in a position to start drawing a graph. This time, we can include the upper and lower bounds for each country, to see the uncertainty about how these figures seem to have changed over time. We'll need to revisit several different techniques we've seen in our previous worksheets: piping, filtering, plotting, using as.numeric, tweaking factor levels, faceting, using viridis, tidying up labels, and using custom themes.

But, in practice, this is how things work.

```
for_final_plot %>%
  filter(country == "United States" |
           country == "Italy" |
           country == "France" |
           country == "United Kingdom") %>%
  ggplot() +
  aes(x = as.numeric(year),
      y = combined,
      colour =
        factor(measure,
               levels = c("Lower", "Rank estimate", "Upper"))) +
  geom_line()+
  facet_wrap(~ country) +
  scale_colour_viridis_d() +
  labs(colour = "Uncertainty",
       x = "Year",
       y = "Voice and accountability percentile") +
  theme_bw()
```

Bonus Exercise

- Find a dataset online on any topic of your choice, and identify a graph you'd ideally like to be able to draw using the data that it contains.
- Have a look at the names of the potential variables - can you draw the graph you were hoping to straight away? If not, can you get it into a format that would allow you to do so?