Lab

Building plots layer by layer with ggplot2

Lab objectives:

- To create plots layer by layer
- To replace values in datasets
- To select data which satisfies a condition (e.g. within a date range).

ggplot2 is a library for the creation of sophisticated plots. It does this by layering detail for the plot. In this lab, we will look at ggplot2's functionality using scatter plots and line plots. You saw histograms, bar and column plots in the previous labs and you will see other plots in later labs.

Type all the examples in this lab and check the results to ensure you understand the concepts being introduced. Exercises are at the end of this lab.

ggplot2

For this lab, you will need library *ggplot2,* which provides functions to create plots. It is part of what is loaded when you use library tidyverse, which we also need, so load the tidyverse library.

```
library(tidyverse)
```

Data

Data must be represented by a data frame (or a tibble).

Note: for the first part of this lab, as in previous labs, we will assume that we have perfect data which we do not need to transform. Towards the end, you will need to replace values in your data.

The datasets to be downloaded have already been manipulated to give the perfect data for our purposes, with the exception of replacing some values (see section later on).

Download the following files from CampusMoodle:

- *consumptionNoOil.csv:* It contains data regarding the consumption of various energy sources since 1990. The dataset contains the year, the Million Ton of Oil Equivalent (MTOE), the estimated MTOE (EstMTOE, before the actual figure was known) and the energy source. Where the date is in the future, MTOE and estMTOE are the same.

- *provenOilReservesZerosT.csv:* it contains data regarding proven oil reserves for European countries.

- *provenOilReserveWEurope.csv*: contains data regarding proven oil reserves for western European countries.

Load the first file and inspect the data to get an idea about the contents of the dataset. You may want to use:

```
summary(consumptionNoOil)
head(consumptionNoOil, 10)
```

ggplot2 basics

To create a plot with *ggplot*, the function needs to be given 2 parameters:

- Data: in the form of a data frame representing the data to be visualised
- Aesthetics mapping: these include colour, size, shape and other attributes.

Layers

A layer is specified as follows:

layer(geom, geom_parameters, stat, stat_parameters, data, mapping, position)

geom (optional): this specifies the geometric object, i.e. the visualisation means of representing the data. For example, if we were using a scatterplot, the geom is "point". Every geom has a set of default settings for statistics and position (see below). Last week you saw histogram, bar and col.

geom_params: describe the parameters for the geom selected. For example, a histogram can have a "fill" colour.

stats (optional): describes the statistics used.

stats_param: describes the parameters used for the selected statistics. For example, the statistics for a histogram is the "bin", and the parameter is the binwidth.

mapping (optional): a set of aesthetic mappings which are described using the aes() function. The x and y axis are mapped.

position (optional): describes the method for adjusting overlapping objects.

There a shortcuts which can be used instead of the full layer specification. These are:

geom_XXX(mapping, data, …, geom, position)

stat_XXX(mapping, data, …, stat, position)

where XXX is replaced by a visualisation method, e.g. for a histogram use

geom_histogram.

Aesthetics

The aesthetics map the variables in the data to the things that we can see in the plot. For example, we can map variables to the x and y axis.

Aesthetics can be set (so that they are constant) or mapped. It is usually convenient to use a variable to store your definitions. For example, we can try

```
p <- ggplot(data= consumptionNoOil, aes(x=MTOE,y=EstMTOE))
```

to define the data (consumptionNoOil) and some aesthetics (MTOE is on the x axis and EstMTOE on the y axis) and we store this in variable p. This does not say which type of plot we want. We can add this by, for example saying that we want a scatterplot and we call p to actually visualise the plot.

```
p <- p + geom_point()
p
```

Note how we have been constructing our plot and saving it on variable "p". We first stated the dataset and the x and y axis variables. We then added the geom to the plot, by adding it to what is already stored in variable "p". Finally we call "p" so that the plot is displayed.

Note that if you follow the pre-defined order you can omit some the keywords ("data=", "x=", y="), so the code below has the same effect as the code above.

```
p <- ggplot(consumptionNoOil, aes(MTOE, EstMTOE))
p <- p + geom_point()
p
```

To set the colour of all points to dark green.

```
p <- p + geom_point(colour= "darkgreen")
p
```

Note how we use variable p to store our visualisation, and we "add" to it.

Column "Source" indicates whether the source is coal, gas or renewables. We could use column "Source" as an additional dimension, by using it for colour. Try

```
p <- ggplot(consumptionNoOil, aes(MTOE, EstMTOE, colour=Source))
p <- p + geom_point()
p
```

We could use column "Source" as an additional dimension, by using it for shape. Try

```
p <- ggplot(consumptionNoOil, aes(MTOE, EstMTOE, shape=Source))
p <- p + geom_point()
p
```

We could use column "Source" as an additional dimension, by using it for both colour and shape. Try

```
p <- ggplot(consumptionNoOil, aes(MTOE, EstMTOE,
     colour = Source, shape=Source))
p <- p + geom_point()
p
```

You will see how colour is use to indicate the sources (oil, gas, coal and renewables) and the legend indicates this. Note that in this case you have plotted 3 variables, using colour as the 3rd dimension.

When we display time-related data, a line plot is more appropriate. If we wanted lines we could use a line geom instead of the point geom. For example, to display Year against MTOE we could have the following code (which is far from ideal!):

```
p <- ggplot(consumptionNoOil, aes(Year, MTOE))
p <- p + geom_line()
p
```

The above code produces a confusing plot which includes 3 different data points per year (one for gas, one for coal and one for renewables). Ideally we would want one line for each of the energy sources. We can use the Source column to colour one line per energy source.

```
p <- ggplot(consumptionNoOil, aes(Year, MTOE, colour=Source))
p <- p + geom_line()
p
```

Groups

Up till now, we have plotted a single "instance", i.e. point geom (scatterplot) with one point per observation. It is possible to plot collective (group) geoms, i.e. where we have several observations. The group geom is used for this.

2 cases:

Multiple groups with one aesthetics: data is separated into groups which are all rendered in the same way.

For example, to plot year vs consumption by energy source.

```
p <- ggplot(consumptionNoOil, aes(Year, MTOE, group = Source))
p <- p + geom_line()
p
```

We could change the colour and size of the line, for example

```
p <- p + geom_line(size = 2, colour="darkgreen")
p
```

Note how it did not involve removing the ggplot command we had already stored in p, just adding to it.

Similarly, if we just wanted points, we would change the geom. Try

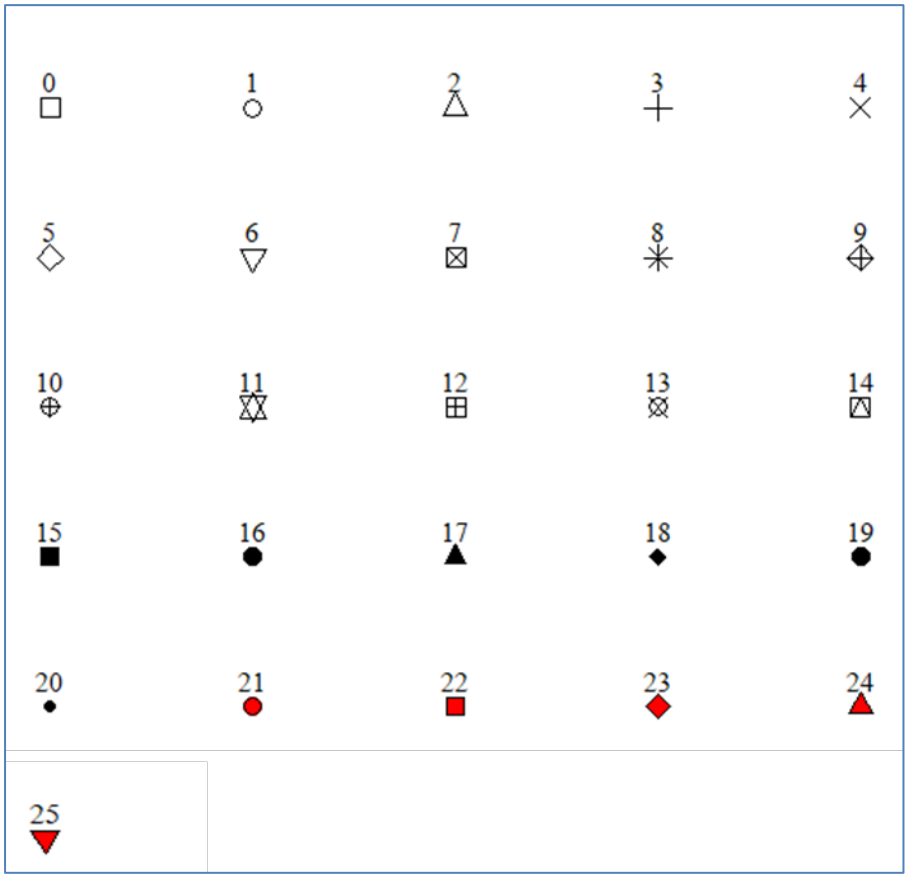
```
p <- ggplot(consumptionNoOil, aes(Year, MTOE, group = Source))
p <- p + geom_point()
p
```

Note that to display the line we have had to re-do variable p as we had a point geom in it. If we simply added a line geom, both geoms would be displayed.

To display both geoms you could try

```
p <- ggplot(consumptionNoOil, aes(Year, MTOE, group = Source))
p <- p + geom_point()
p <- p + geom_line()
p
```

The following points are available in R



Points 21 to 25 can be coloured (shown in red in the figure).

If we wanted to change the shape and size of the geom and add some colour to it we can state it in the geom function. For example, if we want light blue large triangles (size 4) with a red outline, we could just add this information in the geom_point parameter list, i.e. try

```
p <- p + geom_point(shape = 24, colour = "red", size = 4,
      fill = "lightblue")
p
```

While this may be useful for some visualisations, it is not possible to know which line/point belongs to which source type.

Different groups on different layers: used when we want to plot summaries at different levels of detail. To plot all lines as before plus a "summary" line (smooth based on all data per year)

```
p <- ggplot(consumptionNoOil, aes(Year, MTOE, group = Source))
p <- p + geom_line()
p <- p + geom_smooth(aes(group = 1), formula='y~x',
      method = "lm", size = 2, se = F)
p
```

Note that

```
p <- ggplot(consumptionNoOil, aes(Year, MTOE, group = Source))
p <- p + geom_line()
p <- p + geom_smooth(aes(group = Source), method = "lm",
      size = 2, se = F)
p
```

Would create one smooth per line, which is not what we want. You will see "smooths" on the analysis part of the subject.

Replacing values

We will be using dataset ProvenOilReservesZerosT. Load it using

```
ProvenOilReservesZerosT <- read.csv("ProvenOilReservesZerosT.csv",
header=T)
```

The file contains symbol "-" wherever a value is not known (or zero). We could replace these with NA which is the proper way of indicating missing values in R. The following code replaces value – by NA in every column.

```
ProvenOilReservesZerosT |>
      mutate_all(funs(replace(., .== "-", as.double(NA))))
```

Exercises

1. Load the ProvenOilReservesZerosT.csv dataset.
2. Produce a scatterplot of yearly proven oil vs Denmark.
3. Update the plot in (2) so that it uses dark blue squares of size 4 for your points (you may wish to look at lab 2 for the list of symbols which can be used as points).
4. Produce a line plot of yearly proven oil vs Italy. Your line should be of size 2, of orange colour.
5. Produce a scatterplot of yearly proven oil vs United Kingdom. Use colour to emphasize the size of values.
6. Produce a scatterplot of yearly proven oil vs Norway. Use red colour. The points should be triangles and their size should be a factor of the size value for Norway.
7. Load the provenOilReserveWEurope.csv dataset
8. Produce a line plot of year versus Million Tons of Barrels by country.

9. Update your solution to 8, so that all lines are purple.
10. Update your solution to 8 so that each line is a different colour.
11. Produce a line plot of year versus Million Tons of Barrels by country, but also add a summary (smooth)
12. Same as 11 but make each line a different colour.
13. Same as 12, but use coloured points (a colour per country).

Selecting data satisfying a condition

To select only the rows with Year values between 1980 and 1990 and store these in a dataframe called to1990.

```
to1990 <- ProvenOilReserveWEurope |> filter(Year %in% c(1980:1990))
```

Use the summary function to check the data

```
summary(to1990)
```

Heat Map

Below is the code to produce a heat map for the data to1990 dataset created in the previous section. Note the use of the alpha aesthetic for the colour intensity.

```
p <- ggplot(to1990, aes(Year, Country))
p <- p+ geom_tile(aes(alpha=MT.Barrels), fill= "blue")
p
```

Exercise

14. Produce a heat map for years 1990 to 2001. Use red as the colour.