Session 2b

The Data Carpentry (amended and delivered by Chris Oldnall)

October 25th 2023

Usage and Adaptation of Data Carpentry Materials

Most material found in this document has been adapted from the Data Carpentry [https://datacarpentry.org/r-socialsci/] materials, under the creative commons attribution license [https://creativecommons.org/licenses/by/4.0/]. Minor amendments have been made to allow for compatability in order.

Objectives of the session:

- Produce scatter plots, boxplots, and barplots using ggplot.
- Set universal plot settings.
- Describe what faceting is and apply faceting in ggplot.
- Modify the aesthetics of an existing ggplot plot (including axis labels and colour).
- Build complex and customized plots from data in a data frame.

Questions to be able to answer:

- What are the components of a ggplot?
- How do I create scatterplots, boxplots, and barplots?
- How can I change the aesthetics (ex. colour, transparency) of my plot?
- How can I create multiple plots at once?

Plotting with `ggplot2`

`ggplot2` is a plotting package that makes it simple to create complex plots from data stored in a data frame. It provides a programmatic interface for specifying what variables to plot, how they are displayed, and general visual properties. Therefore, we only need minimal changes if the underlying data change or if we decide to change from a bar plot to a scatterplot. This helps in creating publication quality plots with minimal amounts of adjustments and tweaking.

`ggplot2` functions work best with data in the 'long' format, i.e., a column for every dimension, and a row for every observation. Well-structured data will save you lots of time when making figures with `ggplot2`

ggplot graphics are built step by step by adding new elements. Adding layers in this fashion allows for extensive flexibility and customization of plots.

Components

Each chart built with ggplot2 must include the following

- Data
- Aesthetic mapping (aes)
  - Describes how variables are mapped onto graphical attributes
  - Visual attribute of data including x-y axes, color, fill, shape, and alpha
- Geometric objects (geom)
  - Determines how values are rendered graphically, as bars (`geom_bar`), scatterplot (`geom_point`), line (`geom_line`), etc.

Thus, the template for graphic in ggplot2 is:

```
<DATA> %>%
    ggplot(aes(<MAPPINGS>)) +
    <GEOM_FUNCTION>()
```

Plotting Options

ggplot2 offers many different geoms; we will use some common ones today, including:

- `geom_point()` for scatter plots, dot plots, etc.
- `geom_boxplot()` for, well, boxplots!
- `geom_line()` for trend lines, time series, etc.

To add a geom to the plot use the + operator.

Notes on +

The `+` in the `ggplot2` package is particularly useful because it allows you to modify existing `ggplot` objects. This means you can easily set up plot templates and conveniently explore different types of plots

Notes

As you will learn, there are multiple ways to plot the a relationship between variables. Another way to plot data with overlapping points is to use the `geom_count` plotting function. The `geom_count()` function makes the size of each point representative of the number of data items of that type and the legend gives point sizes associated to particular numbers of items.

Barplots

Barplots are also useful for visualizing categorical data. By default, `geom_bar` accepts a variable for x, and plots the number of instances each value of x (in this case, wall type) appears in the dataset.

Adding Labels and Titles

By default, the axes labels on a plot are determined by the name of the variable being plotted. However, **`ggplot2`** offers lots of customization options, like specifying the axes labels, and adding a title to the plot with relatively few lines of code. We will add more informative x-and y-axis labels to our plot, a more explanatory label to the legend, and a plot title.

The `labs` function takes the following arguments:

- `title` – to produce a plot title
- `subtitle` – to produce a plot subtitle (smaller text placed beneath the title)
- `caption` – a caption for the plot
- `...` – any pair of name and value for aesthetics used in the plot (e.g., `x`, `y`, `fill`, `color`, `size`)

Faceting

Rather than creating a single plot with side-by-side bars for each village, we may want to create multiple plot, where each plot shows the data for a single village. This would be especially useful if we had a large number of villages that we had sampled, as a large number of side-by-side bars will become more difficult to read.

`ggplot2` has a special technique called *faceting* that allows the user to split one plot into multiple plots based on a factor included in the dataset. This involves using ‘facet_wrap()’.

`ggplot2` themes

In addition to `theme_bw()`, which changes the plot background to white, **`ggplot2`** comes with several other themes which can be useful to quickly change the look of your visualization. The complete list of themes is available at https://ggplot2.tidyverse.org/reference/ggtheme.html. `theme_minimal()` and `theme_light()` are popular, and `theme_void()` can be useful as a starting point to create a new hand-crafted theme.

The ggthemes package provides a wide variety of options (including an Excel 2003 theme). The **`ggplot2`** extensions website provides a list of packages that extend the capabilities of **`ggplot2`**, including additional themes.

Saving Plots

After creating your plot, you can save it to a file in your favourite format. The Export tab in the **Plot** pane in RStudio will save your plots at low resolution, which will not be accepted by many journals and will not scale well for posters.

Instead, use the `ggsave()` function, which allows you to easily change the dimension and resolution of your plot by adjusting the appropriate arguments (`width`, `height` and `dpi`).

Keypoints

- `ggplot2` is a flexible and useful tool for creating plots in R.
- The data set and coordinate system can be defined using the `ggplot` function.
- Additional layers, including geoms, are added using the + operator.
- Boxplots are useful for visualizing the distribution of a continuous variable.
- Barplots are useful for visualizing categorical data.
- Faceting allows you to generate multiple plots based on a categorical variable.