A Two Page Guide to ggplot2

Andy Grogan-Kaylor (http://www.umich.edu/~agrogan)

February 23, 2019

[Share on Twitter]

1 Why?[1]

A great deal of data analysis and visualization involves the same core set of steps: get some data, clean it up a little, run some descriptive statistics, run some bivariate statistics, create a graph or a visualization. **ggplot2** can be an important part of a replicable, automated, documented workflow for complex projects.

[1] More information can be found here: http://www-personal.umich.edu/~agrogan/ggplot2

Given the fact that we often want to apply the same core set of tasks to new questions and new data, there are ways to overcome the steep learning curve and learn a replicable set of commands that can be applied to problem after problem.[2]

[2] The same 5 to 10 lines of ggplot2 code can often be tweaked over and over again for multiple projects.

2 The Essential Idea Of ggplot2 Is Simple

There are 3 essential elements to any ggplot call:

1. An *aesthetic* that tells ggplot which variables are being mapped to the *x axis*, *y axis*, (and often other attributes of the graph, such as the *color fill*). Intuitively, the aesthetic can be thought of as **what you are graphing**.
2. A *geom* or *geometry* that tells ggplot about the basic structure of the graph. Intuitively, the geom can be thought of as **how you are graphing it**.
3. Other options, such as a *graph title*, *axis labels* and *overall theme* for the graph.

3 Get Started

```
library(ggplot2)  # beautiful graphs

library(ggthemes)  # nice themes for ggplot2
```

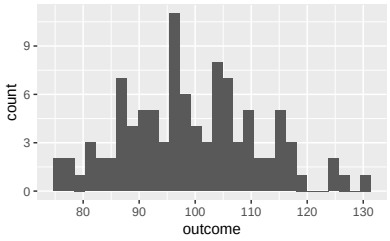
predictor	outcome	group
96.59	97.31	A
90.43	93.97	A
101.9	93.98	A
103.8	108.2	A
114.9	112.4	A
106	104.5	A

Figure 1: Sample Data

4 Some Examples

4.1 One Continuous Variable

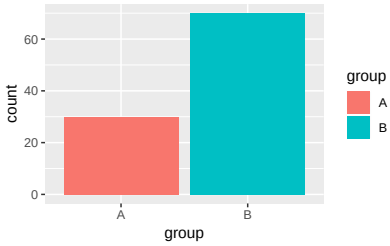
```
# anything that starts with a '#' is a comment

ggplot(mydata, # the data I am using
       aes(x = outcome)) + # the variable I am using
  geom_histogram() # how I am graphing it
```



4.2 One Categorical Variable

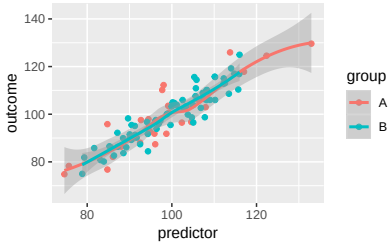
```
ggplot(mydata, # the data I am using
       aes(x = group,
           fill = group)) + # the variable I am using
  geom_bar() # how I am graphing it
```

Changing variables from factor to numeric (e.g. `aes(x = as.numeric(outcome))`), and *vice versa* can sometimes be the simple solution that solves a lot of problems when you are trying to graph your variables.



4.3 Continuous by Continuous

```
ggplot(mydata, # the data I am using
       aes(x = predictor, y = outcome,
           color = group)) + # the variables I am using
  geom_point() + # how I am graphing it
  geom_smooth()
```



4.4 Add Some Options[3]

```
ggplot(mydata, # the data I am using
       aes(x = predictor, y = outcome,
           color = group)) + # the variables I am using
  geom_point() + # how I am graphing it
  geom_smooth() +
  labs(title = "My title",
       x = "title for x axis",
       y = "title for y axis") +
  scale_colour_manual(name = "Group",
                      values = c("#FF0099",
                                 "#0099FF"), # manual colors
                      labels = c("group1",
                                 "group2")) +
  theme_minimal() + # theme
  theme(plot.title = element_text(size = 20, # add to theme
                                  color = "#0099FF"))
```

[3] Notice how use of `scale_ ...` governs both the use of color in the graph below, as well as the legend that is produced in the graph. This graph uses official UNICEF colors to illustrate this idea. The graph below uses `scale...manual...` to manually choose the colors, but there are many other options, particularly when using `library(ggthemes)`. `scale_..._viridis_...` are especially good color palettes.