it is possible to generate attractive looking plots using just base graphics. Although we did confess, that it did take a lot of time and effort. In this post, we repeat the same exercise. Using the dreaded iris data set, we'll first create the default ggplot2 graph, before applying a bit of care and attention.

**The standard ggplot version**

The standard scatter plot is straightforward to create. Load the package

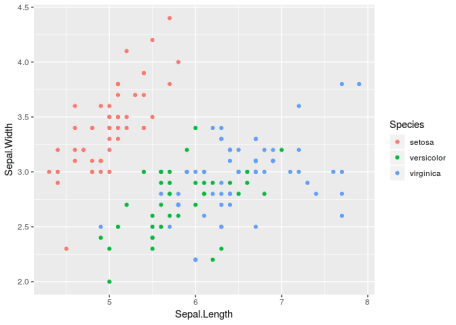
library("ggplot2")

Then create a scatter plot with the wonderful grey background

## ggplot2 even spells colour correctly ;)

ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width)) +

geom\_point(aes(colour = Species))



Unlike the base R offering, the list of possible improvements to this plot is pleasingly short. Basically, it's

* the axis labels (but they come from **our** column headings)
* colours (red & blue aren't the best combination)

So overall, pretty good. Other aspects that could be improved are

* grey background
* direct labels on the points
* starting the x-axis at 4, not 4.2

**Styling the plot using only ggplot2**

Using only ggplot2 (and a little bit of **dplyr** love), we can improve significantly and easily improve the graph. First, we'll capitalise the legend key. I find it easier to manipulate the data directly,

library("dplyr")

iris = mutate(iris, Species = stringr::str\_to\_title(Species))

With the data tweaked, we can get to the serious business of styling the plot. As the plot will contain a number of components it makes sense to create intermediate objects. As the points overlap, we’ll change from geom\_point(), to geom\_jitter(). This geom wiggles the points and allow us to see overlapping points:

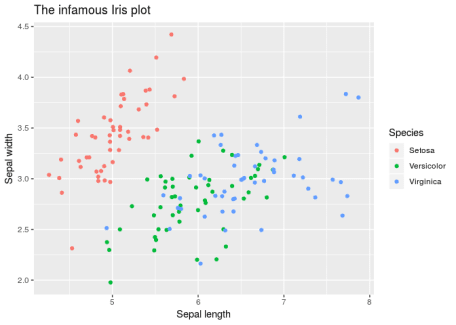
g = ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width)) +

geom\_jitter(aes(colour = Species)) +

xlab("Sepal length") + ylab("Sepal width") + # Improve axis labels

ggtitle("The infamous Iris plot") # Title

g

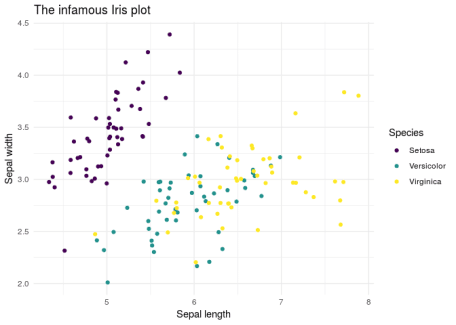
The changes we've made so far would impossible for any package to do for us – how would the package know the plot title? We can now improve the look and feel of the plot. There are two ways of complementary ways of doing this: scales and themes. The ggplot scales control things like colours and point size. In the latest version of ggplot2, version 3.0.0, the Viridis colour palette was introduced. This palette is particularly useful for creating colour-blind friendly palettes

g + scale\_colour\_viridis\_d() # d for discrete

The theme controls elements such as grid lines, fonts, labels. I'm partial to theme\_minimal()

g + scale\_colour\_viridis\_d() +

theme\_minimal()



**The hrbrthemes package**

We don't just have to use the themes that come with **ggplot2**, we can use themes provided by other packages. The **hrbrthemes** packages contain a nice theme called ipsum that’s similar to the minimal theme, but also tweaks the font and allows sub-headings. There is also an associated colour scheme called scale\_colour\_ipsum()`. An additional improvement we’ll make, is to drop the legend and place the text directly on the chart. After loading the package

library("hrbrthemes")

we create a data frame with the label positions

labels = data.frame(x = c(5, 5.3, 7), y = c(4.2, 2.1, 3.7),

Species = c("Setosa", "Versicolor", "Virginica"))

We construct the plot as usual

ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width)) +

geom\_jitter(aes(colour = Species)) +

theme\_ipsum() +

labs(x = "Sepal length", y="Sepal width",

title = "The infamous Iris data set",

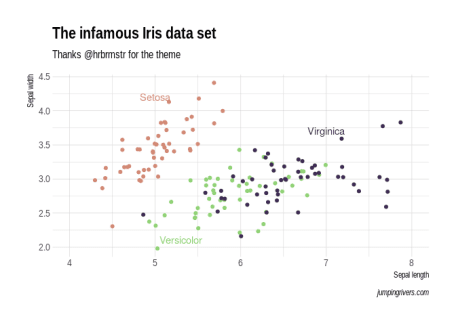
subtitle = "Thanks @hrbrmstr for the theme",

caption = "jumpingrivers.com") +

scale\_colour\_ipsum(guide = FALSE) +

geom\_text(data = labels, aes(x, y, label = Species, colour = Species)) +

xlim(c(4, 8))



Notice we can add data from two data sets onto a ggplot with relative ease.

Thanks for reading, see you next time!

[](https://www.jumpingrivers.com/?utm_source=blog&utm_medium=image&utm_campaign=2018-ggplot2)