Creating high quality figures

Eugene Katsevich

August 28, 2022

This document gives some suggestions on how to create high-quality figures, an essential skill for communicating your results effectively, whether in a paper, talk, or poster. Some of the material is drawn from Chapter 28 of R for Data Science.

1 Sizing

Once you have created a plot in R, you need to export it to include it in your paper, talk, or poster. For example, suppose we have the plot `p` defined as below:

```
test_data <- tibble(x = rnorm(10), y = rnorm(10))
p <- test_data %>% ggplot(aes(x = x, y = y)) + geom_point() + theme_bw()
```

You should save it as a PDF via `ggsave`:

```
ggsave(plot = p,
       filename = "figures-and-tables/test_plot.pdf",
       device = "pdf",
       width = ???,
       height = ???)
```

Here, the question marks should be the width and height of the figure, in inches. Choose these to get a reasonable aspect ratio and a reasonable overall plot size. The aspect ratio (i.e. ratio of width to height) of your plots should be consistent with their content; e.g. box plots are usually relatively narrow, and scatter plots often make sense with equal aspect ratios. The overall plot size should be such that the smallest text in your plots should be roughly the same size as the text in the rest of your slides/paper/poster. Figures 1, 2, and 3 consider the width and length of the figure to be 1 inch, 2.5 inches, and 5 inches, respectively. The medium-sized plot (Figure 2) appears to be the most sensible choice. It usually takes a few tries to find an appropriate size for a figure. If the figures are going into a LaTeX report, I recommend `\includegraphics` using the original scale of the figure you created (i.e. not using commands like `width = 0.8\textwidth`):

```
\begin{figure}[h!]
\centering
\includegraphics{figures-and-tables/test_plot.pdf}
\caption{A test plot.}
\label{fig:test-plot}
\end{figure}
```

This will create consistent font sizes throughout your document.

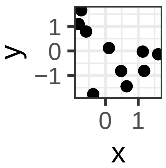

Figure 1: The plot saved as 1in by 1in.

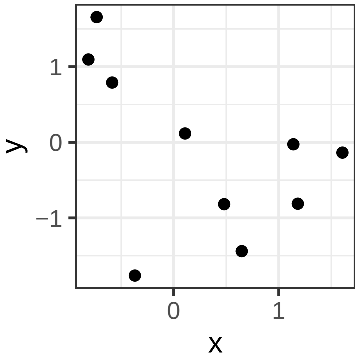

Figure 2: The plot saved as 2in by 2in.

2 Titles

Each plot should include informative axis and legend titles. For example, consider the code below (drawn from R4DS Chapter 28), which produces the plot in Figure 4.

```
# a plot without clear axis and legend titles
p <- mpg %>%
  ggplot(aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme_bw()

# save plot
ggsave(plot = p,
       filename = "figures-and-tables/cars-unlabeled.pdf",
       device = "pdf",
       width = 5,
       height = 3.75)
```

This is a plot of fuel efficiency versus engine displacement for various types of cars, but the axis and legend labels on the plot do not make this very clear. We can easily add informative titles to this plot using `labs`, resulting in Figure 5, which is much easier to understand.

```
# a plot with clear axis and legend titles
p <- mpg %>%
  ggplot(aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
```

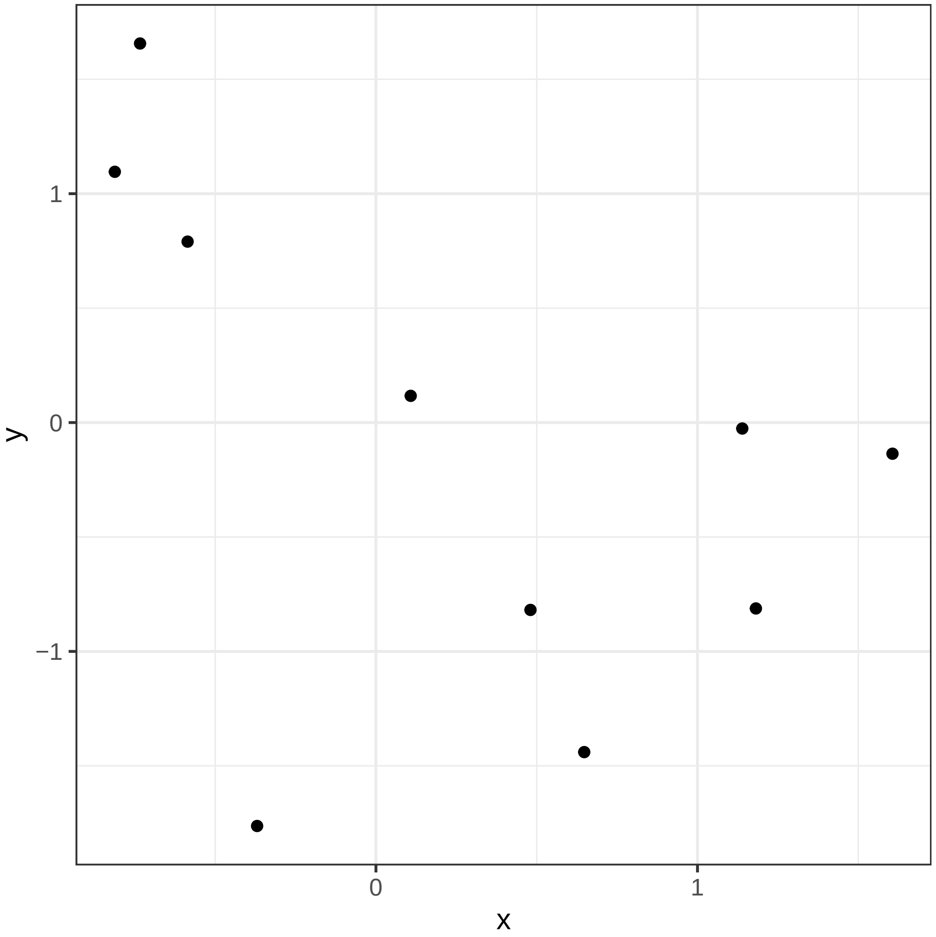

Figure 3: The plot saved as 5in by 5in.

```
  geom_smooth(se = FALSE) +
  labs(
    x = "Engine displacement (liters)",
    y = "Highway fuel economy (miles per gallon)",
    colour = "Car type"
  ) +
  theme_bw()

# save plot
ggsave(plot = p,
       filename = "figures-and-tables/cars-labeled.pdf",
       device = "pdf",
       width = 5,
       height = 3.75)
```

Plots might or might not need overall titles; often the axis titles speak for themselves and the message of the plot can be conveyed in the caption (as in Figure 5.) To add plot titles if necessary, use the `title` argument to `labs()`.

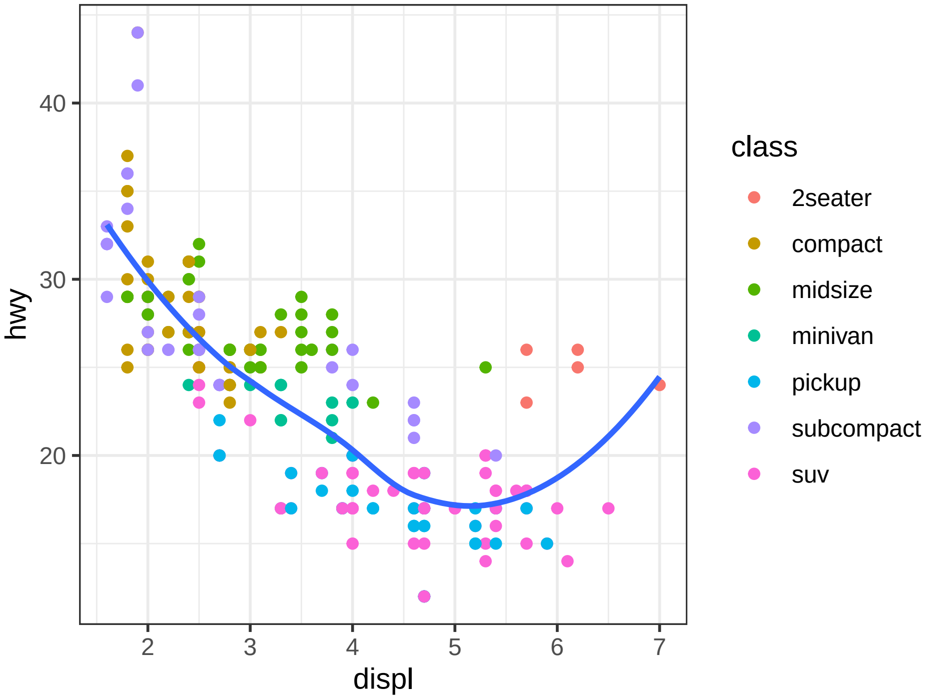

Figure 4: A plot without clear titles.

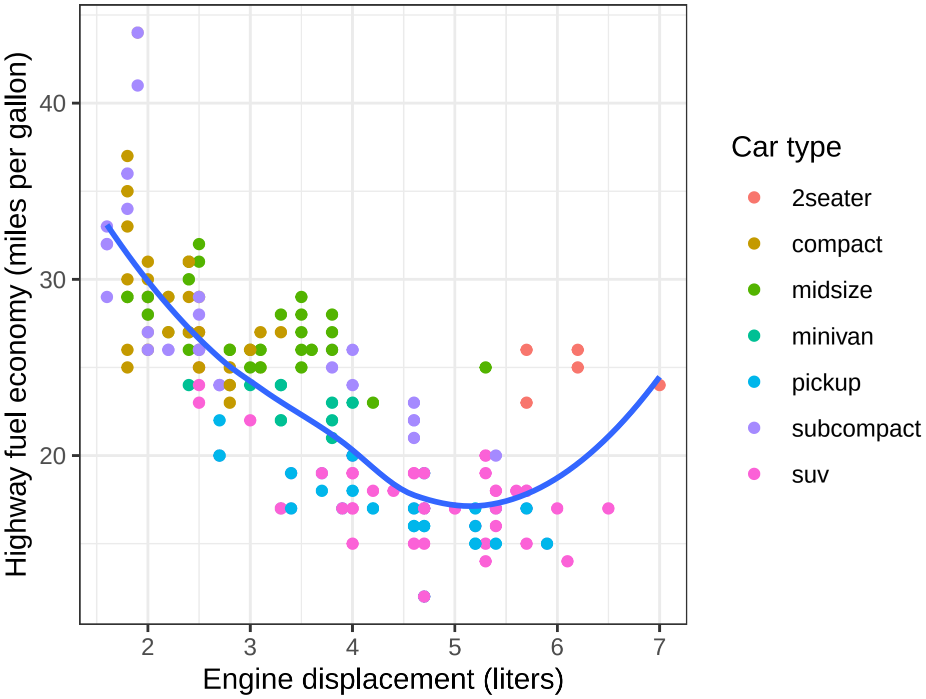

Figure 5: (A plot with clear axis and legend titles). Fuel efficiency generally decreases with engine size; two-seaters (sports cars) are an exception because of their light weight.

If applicable, axis titles should also include the units of measurement, e.g. liters or miles per gallon as in Figure 5. If axis titles involve mathematical formulas, these should be typeset appropriately. The code below (drawn from R4DS Chapter 28) and Figure 6, which it produces, illustrate how to do this. More examples can be found at `?plotmath`.

```r
# a plot illustrating how to include formulas in axis titles
p = tibble(x = runif(10),
       y = runif(10)) %>%
  ggplot(aes(x, y)) +
  geom_point() +
  labs(x = quote(sum(x[i] ^ 2, i == 1, n)),
       y = quote(alpha + beta + frac(delta, theta))) +
  theme_bw()

# save the plot
ggsave(plot = p,
       filename = "figures-and-tables/fig-formulas.pdf",
       device = "pdf",
       width = 2.5,
       height = 2.5)
```

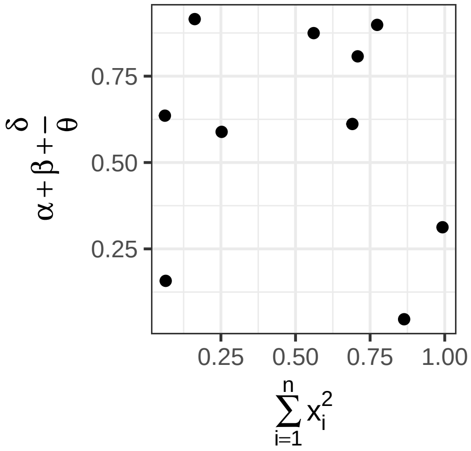

Figure 6: An illustration of using formulas in axis titles.

3 Captions

Figures should have informative captions to help readers understand what information is displayed and how to interpret it.

4 Layout

Sometimes, two or more plots make sense to present together in a single figure. This can be accomplished in two ways. If the different plots convey the same type of information but for different slices of the data, then `facet_grid` and `facet_wrap` are the best way of laying out these plots. For example, the code below and Figure 7 illustrates `facet_wrap` for the `mpg` data used in Figures 4 and 5.

```r
# illustrate how to use facet_wrap to create a multi-panel plot
p = mpg %>%
  filter(class %in%
           c("2seater", "compact", "midsize")) %>%  # select 3 classes of cars
  ggplot(aes(x = displ, y = hwy)) +
  geom_point() +
  facet_wrap(class ~ .) +                          # separate panels per class
  labs(
    x = "Engine displacement (liters)",
    y = "Highway fuel economy\n(miles per gallon)", # line break in axis title
  ) +
  theme_bw()

# save the plot
ggsave(plot = p,
       filename = "figures-and-tables/facet-wrap.pdf",
       device = "pdf",
       width = 5.5,
       height = 2.25)
```

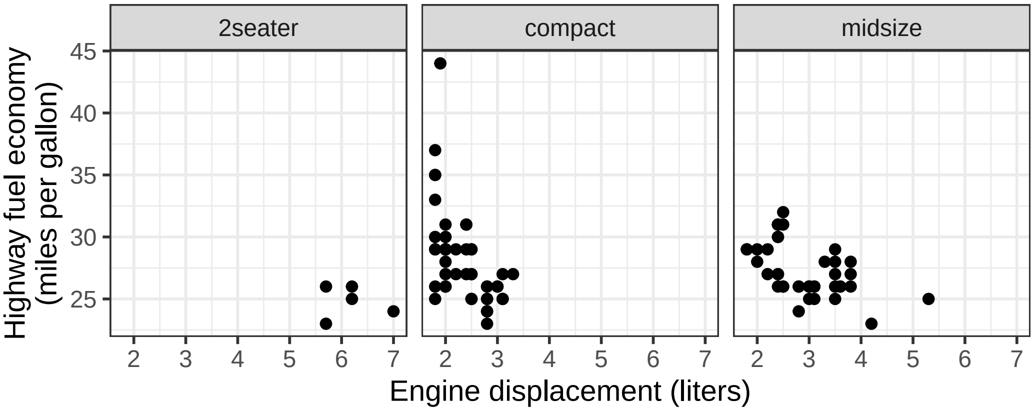

Figure 7: An illustration of using `facet_wrap` to create a multi-panel plot.

If the plots convey different types of information, then they should be created separately and then concatenated together using `plot_grid` from the `cowplot` package. An example is shown below and in Figure 8. Note that the figure caption should reference the subpanels by their labels (in this case, a and b).

```r
# illustration of using cowplot to concatenate multiple plots
library(cowplot)

# first plot: box plot of fuel economy by car type
```

```
p1 = mpg %>%
  mutate(class =                         # re-order car classes by fuel economy
           fct_reorder(class, hwy)) %>%
  ggplot(aes(x = class, y = hwy, fill = class)) +
  geom_boxplot() +
  labs(
    x = "Car type",
    y = "Highway fuel economy\n(miles per gallon)"
  ) +
  theme_bw() +
  theme(legend.position = "none",        # remove legend and x axis text because
        axis.text.x = element_blank())   #  information present in second plot

# second plot: scatter plot of fuel economy versus car type
p2 = mpg %>%
  mutate(class =                         # re-order car classes by fuel economy
           fct_reorder(class, hwy)) %>%
  ggplot(aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  labs(
    x = "Engine displacement (liters)",
    colour = "Car type"
  ) +
  theme_bw() +
  theme(axis.title.y = element_blank())  # remove y axis title because already
                                         #  present in the first plot

# use plot_grid from cowplot to concatenate the two plots
p = plot_grid(p1,
              p2,
              labels = "auto",      # generate labels for subplots
              rel_widths = c(1,2), # specify relative widths
              align = "h")         # how to align subplots

# save the plot
ggsave(plot = p,
       filename = "figures-and-tables/cowplot-demo.pdf",
       device = "pdf",
       width = 5,
       height = 2.5)
```

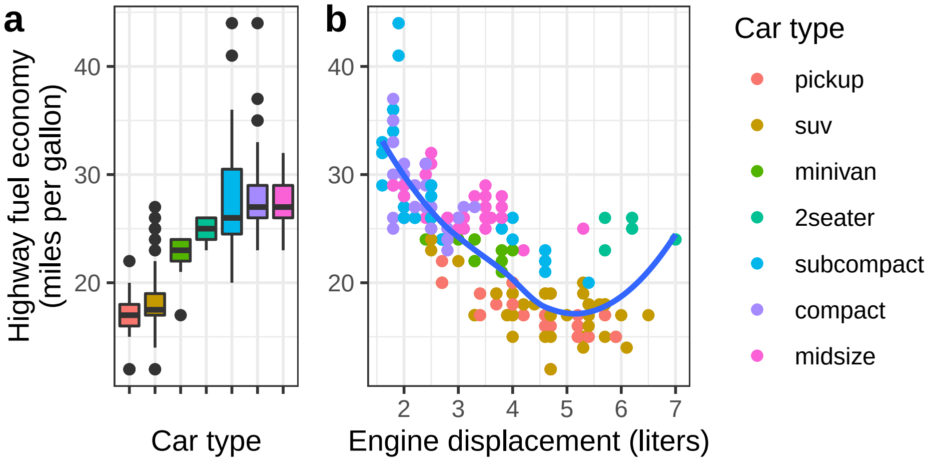

Figure 8: (An illustration of using `cowplot` to create a multi-panel plot.) Relationships between highway fuel economy and car type (a) and engine displacement (b).

5 Further resources

- Chapter 28 of R for Data Science
- Ten Simple Rules for Better Figures (PLOS Computational Biology, 2014)
- How to Create Publication-Quality Figures (by Benjamin Nanes)