Exercise 5: Workflow for publication-ready figures

Éric Marty

May 24, 2022

Below, you will find a workflow and boilerplate R code for generating a publication-ready figure for PLOS journals. The idea is to encode all the figure requirements (fonts, font sizes, figure widths, etc.) as variables outside of your plot, and pass these to the PDF graphics device and the plot. Many journals will accept PDF files for figures. PDF gives you the maximum control over the details of your plot (relative sizes, fonts, etc.), and will always print with no loss of resolution. If the journal requires a TIFF, the PDF can be converted to a high resolution TIFF.

Use one of the figures you have developed in this class. Prepare you figure and export the figure to PDF following the boilerplate code.

Now, suppose your paper has been rejected, and you have decided to submit the paper to a new journal. Choose a journal you would like to submit to, look up their figure guidelines, and modify the boilerplate code to reflect the journal's requirements for fonts, font sizes, and figure dimensions. Export to PDF.

Recommended workflow for publication ready figures:

1. Use the PDF device, setting the figure dimensions, font and default font size in the call to `pdf()`. Ensure all text in your plot (inside the call to `pdf()`) fits the publisher's allowed size range (for example, by specifying a multiple of the default font size using the `cex` property).

Note: R has only 3 built-in fonts: Times, Helvetica and Courier. If you want to avoid installing fonts, use one of these three. See http://blog.revolutionanalytics.com/2012/09/how-to-use-your-favorite-fonts-in-r-charts.html for more on installing fonts and embedding them in pdf's.

2. If a TIFF is required, it will often be required to use "LZW" compression. Not all PDF converters are created equal. The most reliable are the desktop applications GIMP (free) and Photoshop (commercial). Open the saved pdf file in GIMP (Linux, Mac) or Photoshop (Mac, Windows). Export as TIFF with the LZW option checked. See the section **Alternate figure export workflows** for a way to export to tiff directly from R. (Note, this is less reliable.)

Here is an example R script using `pdf()` to draw a figure conforming to PLOS figure guidelines:

```
# Typography
## Font
font.family <- "Times" # Must be allowed by publisher and must be installed on your system.
## Font Sizes
font.sizes <- seq(from = 8, # publisher's minimum point size (points)
                  to = 12, # publisher's maximum point size (points)
                  length.out = 5) # give yourself a range of font sizes to work with
font.size.normal <- mean(font.sizes)
font.scales <- font.sizes/mean(font.sizes)
names(font.scales) <- names(font.sizes) <- c("XS", "S", "M", "L", "XL") # named vector of sizes

# Figure dimensions
figure.widths <- c(min=2.63, page=7.5, column=5.2) # in inches, as defined by publisher
figure.heights <- c(min=1, page=8.75) # in inches, as defined by publisher

# PDF output (PDF Graphics Device)
pdf(
 file = "fig1.pdf", # full path and filename of the file to save
 title = "Figure 1", # displayed in title bar of PDF readers
 width = figure.widths['page'], # full width, in inches
 height = figure.heights['page']*.7, # 70% of full height, in inches
 family = font.family, # defined above
 pointsize = font.size.normal # default (normal) size of text (in points). Defined above.
)

# Put your plot(s) here. Specify a font scale factor of XS, S, M, L, or XL:
plot(0:10, ann=FALSE, cex=font.scales['M'])
legend(cex=font.scales['XS'], ...) # etc.

# close PDF file
invisible(dev.off()) # This line saves and closes the file.
```

Alternate figure export workflows:

The `tiff()` graphic device can be used directly, although this is less reliable than the pdf(), and does not give control over fonts and font sizes.

```
tiff(filename = "fig1.tiff", res = 300, compression = "lzw", height=5.2, width=6, units="in")
[plot code]
invisible(dev.off())
```

There are a number of command line tools (outside of R) that can convert PDFs to TIFF. Of these, ImageMagick, leveraged internally by many R packages, is the best.

Plots made with `ggplot2` can be saved to multiple formats in a single line of code using `ggsave()`. However, the most reliable way to export ggplot figures is to print thh plots within a graphic device:

```
p <- ggplot(mtcars, aes(mpg, wt)) +
  geom_point()

pdf("mtcars.png")
print(p)
dev.off()
```