Guidelines and Tips for Making Figures

Adam Morgan
adam.morgan@mail.utoronto.ca

February 5, 2024

1 Introduction

The effective presentation of figures is a critical skill for the modern scientist, one that requires a huge amount of practice to perfect. In this document included I've some concrete suggestions of software packages you may find useful to help make your figures. More importantly, I've outlined general rules to follow when preparing images to supplement mathematical/scientific discussions. I conclude by drawing a few example pictures to bring the guidelines to life.

2 Suggestions of Free Plot-Making Software

These packages are quite popular and well-documented, so with some Googling you can quickly figure out how to do pretty much anything you want with either of them (for this reason, I won't give you a comprehensive tutorial on these packages here).

Disclaimer: The following packages represent my own personal preferences and do not constitute an exhaustive list of high-quality, free plotting software.

- TikZ (including pgfplots): a nice way to produce figures from TeX files. This is what I used to make the figures in this document. See

  `overleaf.com/learn/latex/Pgfplots_package`

  to learn the basics. Pretty easy to use but can run slowly if you're making lots of pictures so be warned. Has the advantage of coming with most installations of LaTeX.

- Matplotlib (see `matplotlib.org`): a Python library that is very versatile, very elegant, and very easy to use. Also an industrial favourite. I think Matplotlib comes with NumPy, so if you have a version of Python from the last few years you probably have it on your computer already.

- MATLAB (see `mathworks.com`): a very user-friendly pay-to-use software for all types of scientific computing. MATLAB's plotting package can do essentially anything Matplotlib does. If you're a student, you can probably get MATLAB for free or at a big discount through your institution. There are also free MATLAB alternatives like Octave, but I haven't used these much and so can't vouch for them.

3 Guidelines for Making Figures

- **The Golden Rule**: When looking at one of your figures always ask yourself, "would I understand and appreciate this picture if I saw it in a book or a research article?"... if your answer is "no" then the figure is not finished.
- Make sure every figure tells a story. Think of *why* a piece of information must be described visually, and make sure your figure is focused on conveying that information.
- Label axes clearly.
- Label different curves on the same axes clearly.
- Any points in the plane that are important to the story of the picture should also be labelled.
- Include as many tick marks on your axes as necessary to communicate the big idea of your picture. Sometimes none at all is OK.
- Make sure the font size on all your labels is appropriate.
- Use concise captions. Don't be afraid to defer discussion to the body of the text.
- A figure shouldn't be so densely packed that there's only a small amount of empty white space: such space gives our eyes a rest. Too much white space, however, has no use and can often distract the reader from the important point of an image.
- If your picture includes a legend, make sure it does not cover any vital part of the image.
- Even very good software like Matplotlib sometimes defaults to saving pictures with poor resolution. So, while you see a beautiful image when you run your code, it ends up looking blurry when you add it to a document! The fix for this is to manually change the resolution of the image, which is very easy. You want to Google something like "how to change dpi in [insert software here]" (dpi stands for 'dots per inch'). I find that a dpi of 600 is usually pretty OK.

- **Choose colours carefully**.
  - Many people cannot distinguish certain pairs of colours, and when designing a figure you should (to the best of your ability) make sure it is accessible to such readers. For a demonstration of the importance of colourblind-accessibility and several examples of colourblind-friendly palettes, see

    `davidmathlogic.com/colorblind`.
  - Additionally, even those with full-colour vision may want to print off your figure in black-and-white (BW). So, the figure should still be understandable to these people too.
  - In general, I find the default colours in most software packages (including Tikz and Matplotlib) are not only non-accessible: they look terrible too. Fortunately, using alternatives is very easy. For pgfplots/TikZ, you can select from a huge list of colours available at `latexcolor.com`: just paste the colour definition into your TeX file and it's ready to be used. For Matplotlib (if you use a new-ish version), you can use any colour from the XKCD colour survey. See `xkcd.com/color/rgb` for a list of these colours and their names. To use the XKCD colours in MATLAB, you'll need to download the package at

    `mathworks.com/matlabcentral/fileexchange/46872-intuitive-rgb-color-values-from-xkcd`.
  - Try to stick with matte or pastel colours when possible. These are generally the most pleasing (or, if you're a pessimist, the least migraine-inducing) colours to look at.
  - When including different curves on the same axes, make them different colours and give them different styles (solid, dashed, dotted, etc.).
  - If you want to include a filled contour plot, don't use the "jet" colourmap. In Matplotlib, some of the built-in colourmaps are better, but most of the time I stick to the cmocean colourmaps (see `matplotlib.org/cmocean`) which must be downloaded separately (if you use Python, `pip install cmocean`).

4 Good and Bad Examples

Let's apply the above guidelines to draw a nice, clear plot illustrating the behaviour of the heat kernel

$$S(x,t) \doteq \frac{1}{2\sqrt{\pi\kappa t}} \exp\left(-\frac{x^2}{4\kappa t}\right)$$

as time evolves. This plot is made using pgfplots. I will provide you with the raw TeX file for this document so you can see the code yourself (make sure you import TikZ and pgfplots in the preamble).

Figure 1 is drawn according to the guidelines above: labels are clear, the curves are distinguishable by both colour and style, the colours in the picture are muted and do not hurt the eyes (they are custom colours from `latexcolor.com`), and the picture is not too cramped (there is just enough white space). All of this makes it easy to see the mathematical idea being illustrated in the picture, namely that the heat kernel decays in amplitude and spreads out as time elapses; further, this spread becomes slower over time.

However, seeing some examples of bad pictures may make the ideas stick in your head better. Figures 2 and 3 plot the exact same information as Figure 1 (they even feature the same legend and labels). Both these new figures are much less effective than the original one:

- Figure 2: looking at this picture is painful! All three lines are solid so they couldn't be distinguished in a BW printed version, the colours are too bright and hurt the eyes (these are the default TikZ blue, yellow, green), and in places the green and yellow are difficult to distinguish even by a person like me with full-colour vision.
- Figure 3: while this time the curves look good, the axes are all messed up: the x-axis is asymmetric for no good reason, and the y-axis goes up and down way too much. This figure is 75% useless white space. The reader might be tricked into thinking there is some hidden information in the far regions of the image they need to focus on, otherwise why would the image's author format it so strangely?

Hopefully, you have learned that a few easy choices can really make the difference in turning a horrible figure into a pleasant one.

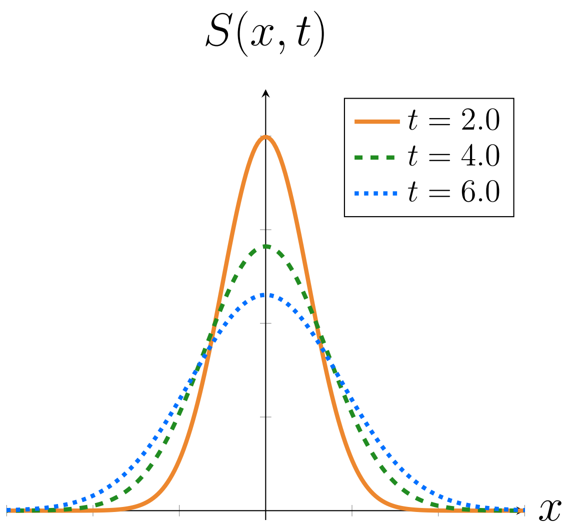

Figure 1: A nice plot illustrating the evolution of the heat kernel. Example of a good “real” caption for this figure: Plot of the heat kernel $S(x,t)$ at various times t for $\kappa = 1/4$. As time elapses, the maximum over x becomes smaller, and the width of the curves increases more slowly.

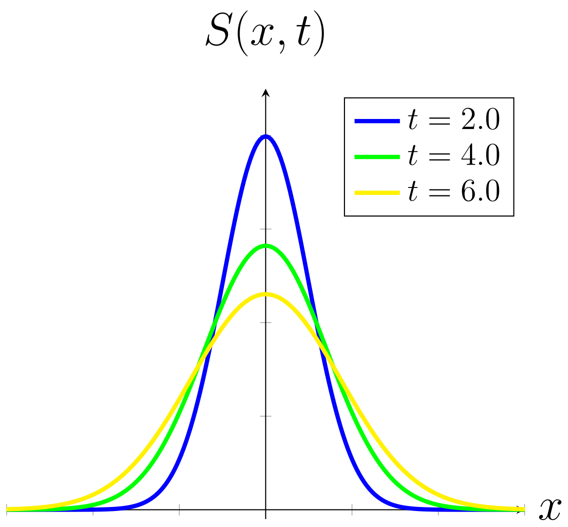

Figure 2: A bad version of Figure 1. Do not make pictures like this!

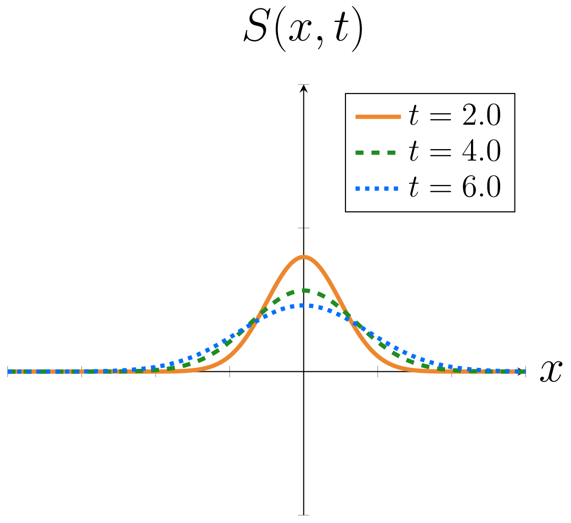

Figure 3: Another bad version of Figure 1. Do not make pictures like this!