A Guide to Preparing Figures

Kunihiko Taira

University of California, Los Angeles

PREPARED BY KUNIHIKO TAIRA FOR HIS RESEARCH GROUP AT UCLA.

Originally created: May 16, 2020; Updated: May 9, 2024

1
Figures

A picture is worth a thousand words.

Figures serve a very special role in the conveyance of scientific messages to the readers. The amount of work and care that goes into the preparation of figures can reward not only the readers to understand the study better but also the writers by facilitating the preparation of the paper.

Having been a writer, reader, reviewer, and editor of many scientific papers, I can say with confidence that most papers with carefully and beautifully prepared figures are often those that have well-organized storylines that deliver the scientific message in an impactful manner. The texts and figures are complementary in a paper and neither of them can be lacking to form a well-prepared manuscript. In this guide, I will discuss how you can prepare figures that are effective in analyzing the data and delivering the scientific message. This guide also provides examples of figure preparation with sample codes in Matlab at the end of each subsection. I have added detailed explanations on why a specific style can be chosen to enhance figures. The goal here is to provide choices to the readers to best present their data to the audience.

What follows hopefully can serve as a simple reference for figure preparation but is not the only way to compile clean effective figures. They come from my personal experience[1] as a researcher. I hope the readers can add their own spices[2] to create beautiful graphics for their papers.

[1] And yes, therefore the guidelines may be biased.
[2] But don't overspice!

1.1 *Basic plotting*

A nicely compiled plot not only conveys the important insights uncovered from your data but also facilitates the delivery of your scientific story. The level of care you put into figures also reveals to the readers how careful of a researcher you are. Simply put: nice figures are wonderful but ugly figures are turn-offs.

Let us start our discussion by considering a simple two-dimensional plot. Say we want to plot a curve. Take $y = \sin(x)$ over $0 \leq x \leq 2\pi$ for example. Here, we use Matlab to generate the plots.[3]. Using the default plotting setup, we obtain the figure 1.1. When you view the default plot on your computer screen, it looks nice (well... maybe...). However, when you embed this figure in a document, it becomes a crime! The labels are so small that they are illegible and will likely irritate readers with bad eye sights[4]. Moreover, the curve ends prematurely before reaching the right end of the plot.

[3] The programs used to generate these plots can be found at the end of each subsection

[4] Including myself, as I'm feeling the age these days.

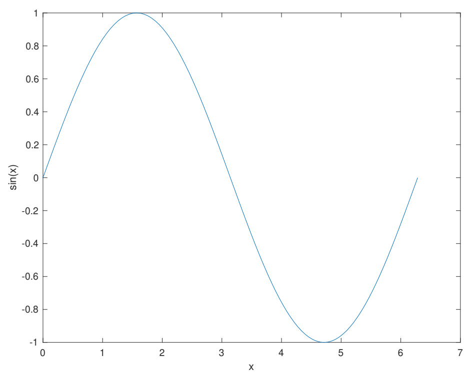

Figure 1.1: A bad two-dimensional plot. This constitutes a crime in my opinion.

Now, we can improve the plot by increasing the font sizes and changing the limits of the x axis to cover 0 to 2π. The improved plot is shown in figure 1.2, which provides a nice representation of the sinusoidal function with legible labels. Make sure that the font size of the labels is appropriate. I recommend the font size used in figures to be slightly smaller than that for the main text. Figures with fonts that are too large or too small look very funny and unprofessional. Another point to consider is the thickness of lines used in the plot. If lines are too thin, they can be hard to view.[5]

[5] In the sample code, I use a Matlab command called `subplot` to make the plot smaller but this has the effect of making the line thicker and fonts larger relative to the overall plot.

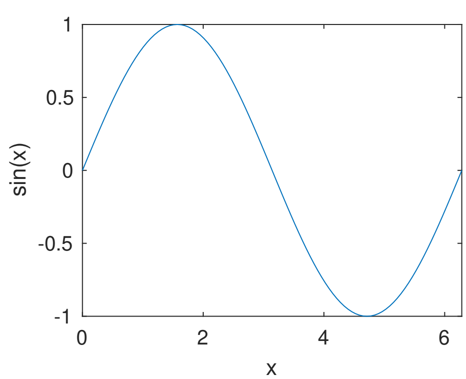

Figure 1.2: An improved two-dimensional plot. It still looks amateurish.

While figure 1.2 looks better than figure 1.1, it is not yet up to the standard of professional publications. We can still improve our plot with a bit of polishing, as shown in figure 1.3. First, we can use fonts that match the text and change the x axis labels in increments of $\pi/2$ to highlight the extrema of the sinusoidal function. We can also add grid lines to facilitate the identification of the extrema and roots.

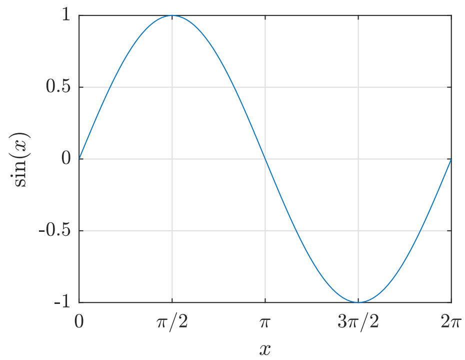

Figure 1.3: A nice two-dimensional plot, which looks much better than the above two plots. Now, I'm happy!

As we can clearly see from comparing the last plot with the previous two plots, the last one is significantly nicer and captures the basic features of the sinusoid concisely. The careful display of results can help you tremendously to convey your scientific story. The readers will appreciate this, too.

```
function plot2d

close all

x = linspace(0,2*pi,101);
f = sin(x);

figure(1)
plot(x,f)
xlabel('x'), ylabel('sin(x)')
exportgraphics(gca,'fig1-01.eps')

figure(2)
subplot(2,2,1)
plot(x,f)
xlabel('x'), ylabel('sin(x)')
xlim([0 2*pi])
exportgraphics(gca,'fig1-02.eps')

figure(3)
subplot(2,2,1)
plot(x,f)
grid on
xticks([0 0.5 1 1.5 2]*pi)
set(gca,'TickLabelInterpreter','latex')
xlabel('$x$','Interpreter','latex')
ylabel('$\sin(x)$','Interpreter','latex')
xticklabels({'$0$','$\pi/2$','$\pi$','$3\pi/2$','$2\pi$'})
exportgraphics(gca,'fig1-03.eps')

end % function plot2d
```

1.2 *Attention to details*

Next, let us consider another example where paying attention to details makes the plot easier to understand. In this example, we have two experimental data sets and two curve fits in the forms of Bessel functions[6]. Plotting the two noisy data sets and the corresponding curvefits using basic settings in Matlab generates figure 1.4. Because this plot is generated with the default setting for viewing data on the screen, it is not optimized for publications. For this reason, figure 1.4 is simply not presentable, analogous to the lousy figure 1.1.

[6] This is a made-up example based on Bessel functions $J_\nu(z)$ of the first kind with $\nu = 1$ and 3. Noise is added to mimic experimental data sets.

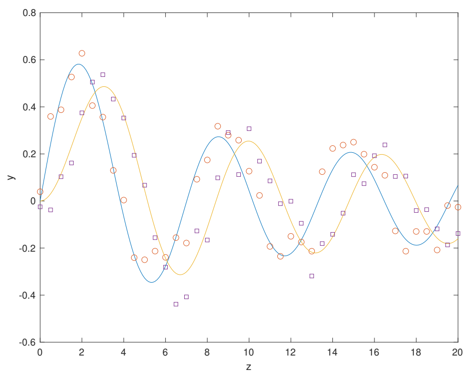

Figure 1.4: Another bad plot (crime). This time for simulated experimental data and their curvefits.

We can tweak some details in figure 1.4 to improve the presentation of the data. First, let us increase the size of the labels and symbols (○ and □). Moreover, we can match the colors of the noisy data and its curvefit for each case. Here, the data fitted with $J_1(z)$ and $J_3(z)$ are shown in blue and red, respectively, in the revised figure 1.5. A legend is also added to identify each of the data and curves. While this figure is better than the previous one, there is an issue with the legend blocking a significant part of the plot.

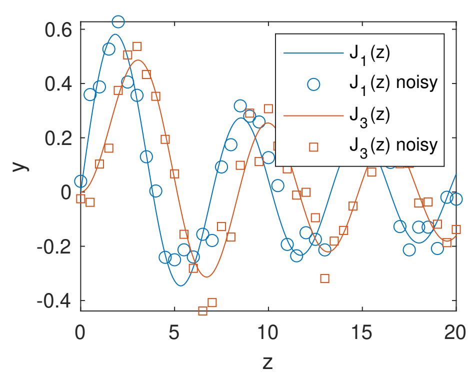

Figure 1.5: A revision of figure 1.4 by using the same color for each case. I'm still not happy.

We can further improve the presentation of the data, as shown in figure 1.6. The legend is replaced with annotations of $y = J_1(z)$ and $y = J_3(z)$ with matching colors. Since the amplitude of oscillations for

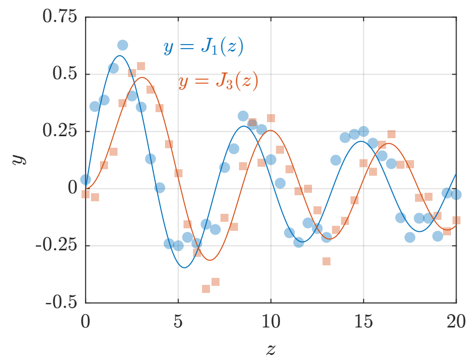

Figure 1.6: Revision with attention to details beyond figure 1.5. This is presentable.

$J_1(z)$ is larger than $J_3(z)$, we can make sure to position $y = J_1(z)$ above $y = J_3(z)$. We also placed these annotations to be close to the respective curves for ease of identification. Another major change made in this revision is the way we plot the data with filled symbols. If the focus of the discussion should be placed on the curve fits, we can soften the color tones of the noisy data sets by using transparency. Furthermore, fonts were changed and grid lines were added to improve readability and presentation, as we have done so for figure 1.3.

```
function plot2d_details

close all

z  = linspace(0,20,201);
zn = linspace(0,20,41);

rng(4000); % fix random numbers
J1  = besselj(1,z);
J1n = besselj(1,zn) + 0.07*randn(size(zn));
J3  = besselj(2,z);
J3n = besselj(2,zn) + 0.07*randn(size(zn));

% Default plot
figure(1)
hold on
plot(z, J1)
plot(zn,J1n,'o')
plot(z, J3)
plot(zn,J3n,'s')
xlabel('z'), ylabel('y'), box on
exportgraphics(gca,'fig2-01.eps')

% Define shades of blue and red
blue = [      0    0.4470    0.7410];
red  = [0.8500    0.3250    0.0980];

figure(2)
subplot(2,2,1)
hold on
plot(z, J1,      'Color',blue)
plot(zn,J1n,'o','Color',blue)
```

```
plot(z, J3,      'Color',red )
plot(zn,J3n,'s','Color',red )
xlabel('z'), ylabel('y'), box on
legend('J_1(z)','J_1(z) noisy','J_3(z)','J_3(z) noisy')
exportgraphics(gca,'fig2-02.eps')

figure(3)
subplot(2,2,1)
hold on
scatter(zn,J1n,30,'filled','o', ...
       'MarkerFaceAlpha',3/8,'MarkerFaceColor',blue)
scatter(zn,J3n,40,'filled','s', ...
       'MarkerFaceAlpha',3/8,'MarkerFaceColor',red)
plot(z,J1,'Color',blue)
plot(z,J3,'Color',red )
set(gca,'TickLabelInterpreter','latex')
set(gca,'defaultTextInterpreter','latex')
xlabel('$z$'), ylabel('$y$')
box on, grid on
text(5,  0.47,'$y=J_3(z)$','Color',red )
text(4.2,0.62,'$y=J_1(z)$','Color',blue)
axis([0 20 -0.5 0.75])
yticks(-0.5:0.25:0.75)
exportgraphics(gca,'fig2-03.pdf')

end % function plot2d_details
```

1.3 *Contour plots and colormaps*

When you present your results in a figure, colors can help you with explaining the results tremendously. Let us take a look at figure 1.7. Here, we show a variety of contour plots for a function[7] $f(x,y)$ over a spatial domain[8] $(x,y) \in [-3,3] \times [-3,3]$. The top row of figure 1.7 shows plots using line contours. The middle row uses filled contours. The bottom row superposes the contour lines over the filled contour plots. The left, middle, and right columns of figure 1.7 present the contours with different colormaps.

[7] This function is an example from Matlab called `peaks`.

[8] When you plot data over a physical space, I recommend using a 1-to-1 aspect ratio for x and y. Otherwise, the plot will stretch the contour over space in an unphysical manner. You wouldn't want street maps stretched only in one direction, would you?

First, let us look at the line contours at the top of figure 1.7. The use of lines clearly displays the contour profiles. The contour curvature and the area encircled by the curves can be easily captured. The filled contours in the middle row provide a vivid visual display of the contours. However, intricate details on the contour distribution may be difficult to extract from these plots. If we overlay the contour lines on top of the filled contours, as shown in the bottom row, the readers can benefit from both the line contours and the filled contours. The images can provide nice graphic displays of the contours that are easy to understand with the ability to precisely capture the profile of $f(x,y)$ which are shown in black.

In this example of figure 1.7, we chose three different colormaps; the jet (left), parula (middle), and red-white-blue (right). The jet used to

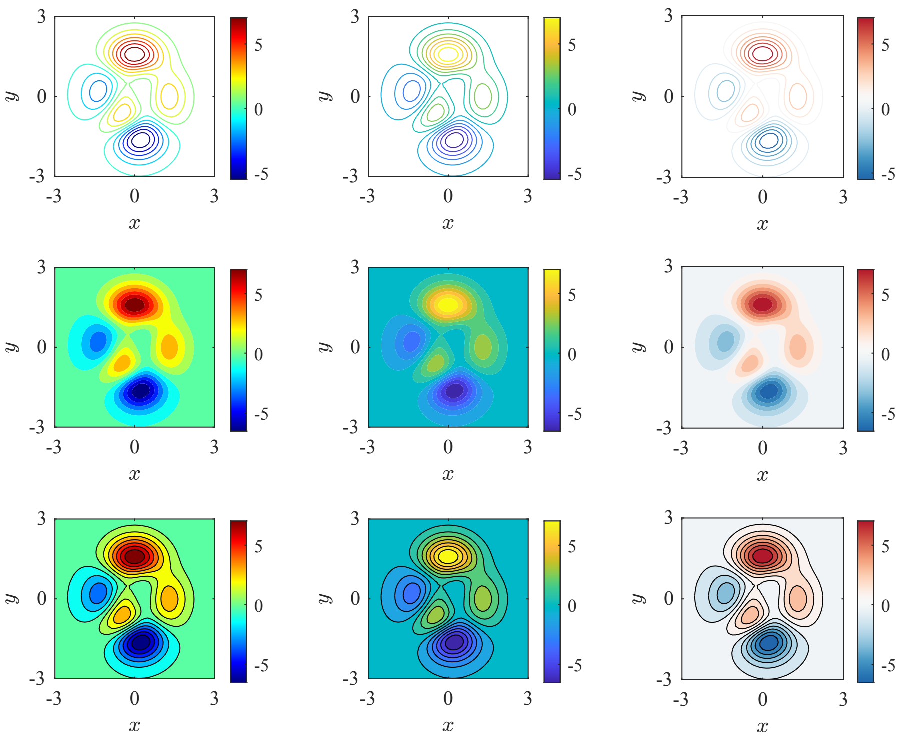

Figure 1.7: Standard colormaps and contour plots.

be the default colormap in Matlab. Recently, Matlab switched to using parula as its default colormap. Both are visually easy to distinguish colors. However, the parula colormap is designed to be colorblind-friendly as discussed below in Section 1.4. For this particular function that is being plotted, the distribution of the functional value of $f(x, y)$ exhibits fluctuations about zero. If our intention is to show high and low values of the function, we should question why we provide any color to the background value of zero. Assigning a color to the background implicitly highlights the background. In this case, I recommend using a white background for the $f = 0$ contour level and assigning red to high and blue to low values. This red-white-blue colormap[9] conveys to the audience that the white area corresponds to the background and highlights the red (high) and blue (low) regions more vividly than the jet and parula[10] colormaps.

[9] Diverging colormap; see MathWorks File Exchange, `othercolor` [1]

[10] Sequential colormap

We show in figure 1.8 the contour plots using different colormaps to examine the way we perceive the data. Used here are the standard colormaps in Matlab as well as those from `othercolor` from Matlab File Exchange [1]. The package `othercolor` provides a larger number of colormap selections. Here, we do not overlay the contour lines. The top row shows the contour plots of $f(x, y)$ using jet (left), hsv (middle), and grayscale (right). Jet and hsv have been widely used in the past but can provide a false sense of profiles in some cases. Parula used

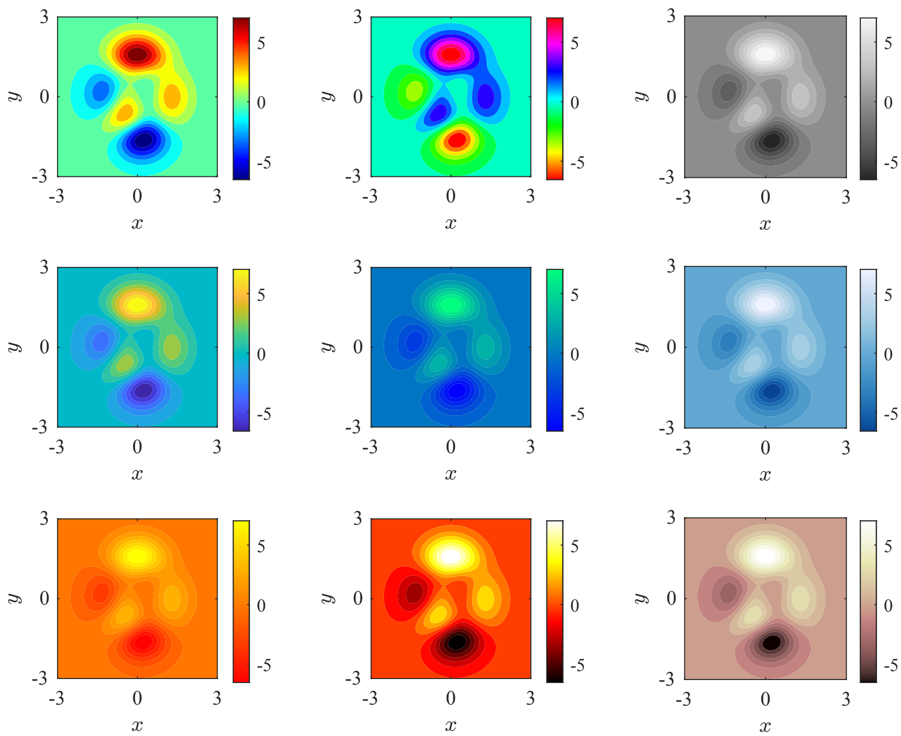

Figure 1.8: Additional colormaps for contour plots [1] .

in middle-left plots can address such issues. In the middle, we used cold colors with shades of blue. These colormaps have calming effects and can be used for cold temperature distributions, for example. In contrast, we use warm colors with shades of red in the bottom row. Such colormaps can be used to visualize hot temperature distribution. Choice of colormaps can be used to visually convey the properties of the plotted variable beyond simple numbers.

```
function contour_color

figure(10)

subplot(3,3,1), example_plot, formatplot
colormap(gca,'jet')

subplot(3,3,2), example_plot, formatplot
colormap(gca,'parula')

subplot(3,3,3), example_plot, formatplot
colormap(gca,flipud(othercolor('RdBu9')))

subplot(3,3,4), example_plot2, formatplot
colormap(gca,'jet')

subplot(3,3,5), example_plot2, formatplot
colormap(gca,'parula')
```

```
subplot(3,3,6), example_plot2, formatplot
colormap(gca,flipud(othercolor('RdBu9')))

subplot(3,3,7), example_plot3, formatplot
colormap(gca,'jet')

subplot(3,3,8), example_plot3, formatplot
colormap(gca,'parula')

subplot(3,3,9), example_plot3, formatplot
colormap(gca,flipud(othercolor('RdBu9')))

exportgraphics(gcf,'contours1.pdf')

%%%

figure(20)

subplot(3,3,1), example_plot2, formatplot
colormap(gca,'jet')

subplot(3,3,2), example_plot2, formatplot
colormap(gca,'hsv')

subplot(3,3,3), example_plot2, formatplot
colormap(gca,flipud(othercolor('Greys7')))

subplot(3,3,4), example_plot2, formatplot
colormap(gca,'parula')

subplot(3,3,5), example_plot2, formatplot
colormap(gca,'winter')

subplot(3,3,6), example_plot2, formatplot
colormap(gca,flipud(othercolor('Blues7')))

subplot(3,3,7), example_plot2, formatplot
colormap(gca,'autumn')

subplot(3,3,8), example_plot2, formatplot
colormap(gca,'hot')

subplot(3,3,9), example_plot2, formatplot
colormap(gca,'pink')

exportgraphics(gcf,'contours2.pdf')

end % function contour_color

function example_plot
    [x,y,f] = peaks;
    contour(x,y,f,13), axis equal
    colorbar
end

function example_plot2
    [x,y,f] = peaks;
    [C,h] = contourf(x,y,f,13); axis equal
    set(h,'LineColor','none')
    colorbar
end
```

```
function example_plot3
    [x,y,f] = peaks;
    contourf(x,y,f,13), axis equal
    colorbar
end

function formatplot
    set(gca,'fontname','times')
    xticks([-3 0 3])
    yticks([-3 0 3])
    axis equal
    xlabel('$x$','Interpreter','LaTeX')
    ylabel('$y$','Interpreter','LaTeX')
end
```

1.4 *Colorblind friendly colormaps*

There is something else that we should consider as we prepare our plots. That is how our choice of colors is perceived by color-blind readers. Approximately 8% of males and 0.4% of females of the populations with Northern European ancestry have some type of the red–green colorblindness (other populations have a lower rate of incidence). There is also the blue-yellow colorblindness with less than 0.01% for the whole population [2]. Those affected by colorblindness are not a negligible portion of the readership. While it may be challenging to choose a color scheme that can best serve your plotting needs and be easily distinguishable by all readers, I believe it is important to understand how our plots are viewed by those with color blindness.

The way colormaps are perceived by color-blind readers is shown in figure 1.9 [3]. It can be seen that some choices of colors may be difficult to distinguish, particularly shades of red and green. For colorblind-friendly colormaps, we can use shades of dark blue and yellow. Matlab has switched their default colormap from jet to parula in 2014 to adopt a perceptually uniform color gradient. This parula colormap is also colorblind friendly and was used in the middle column of figure 1.7. We can also use different lines and symbols to differentiate cases to supplement color choices.

1.5 *Gallery of plots*

The way you can graphically present your results is only limited by your imagination. Let us provide some plots in figure 1.10 to stimulate some constructive and creative visualizations. The graphics presented in figure 1.10 are actual plots used in publications and presentations from our research group [4, 5, 6, 7, 8].

Colors and symbols can be used to differentiate cases discussed in

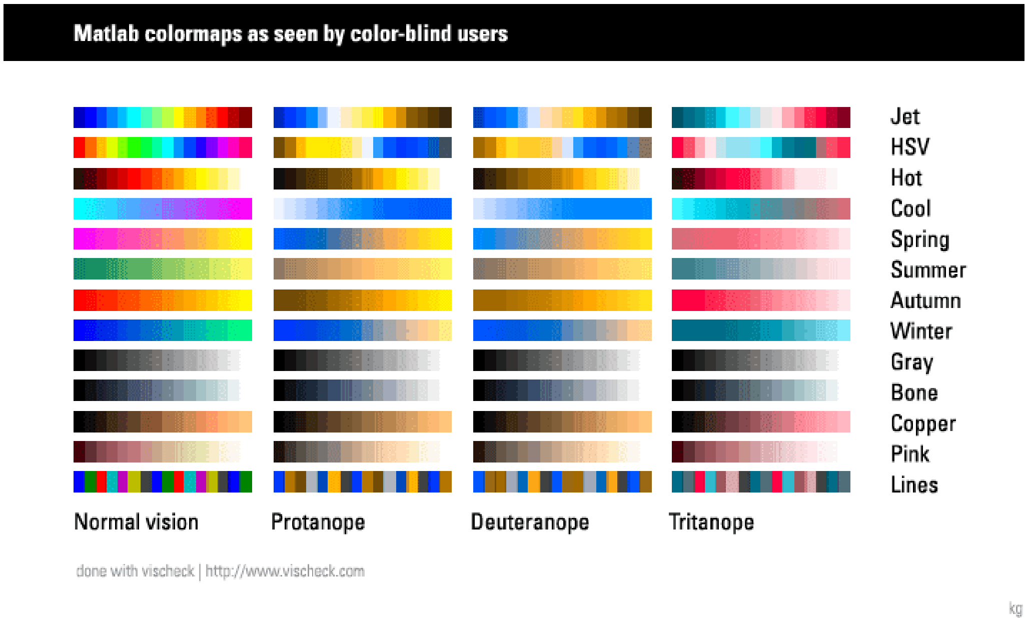

Figure 1.9: Standard Matlab colormaps seen by colorblind readers [3].

the paper as shown in figure 1.10(a) and (b). For figure 1.10(a), the chosen colors for the shown cases are used consistently throughout the paper to easily identify results associated with those cases. In figure 1.10(b), solid and dashed lines are used effectively to highlight the agreement between the two cases. The results in this plot are used to validate a certain computational setup. A red dashed line is selected such that the blue line behind it will appear. The insertion of a zoomed plot also serves as an excellent way to illustrate the nice overlap between the two curves with finer details.

Sometimes, you want to compile more than a few cases into a single plot to display a trend in your data. In such a case, waterfall plots can be helpful, as shown in figure 1.10(c). In this plot, the shift in the spectral peaks is displayed nicely over a parameter space of St and α.

While we have the power to use color at our leisure, sometimes the basic black-and-white color scheme can be simple yet very effective. Shown in figure 1.10(d) are data points over a two-dimensional parameter space (Re_θ and M_∞). Open and filled symbols are used cleanly to indicate which cases are stable and unstable. By also using gray shading in the background, readers can quickly demarcate the two different regions (stable vs. unstable).

Now, we are not limited to showing a single plot at a time. We can combine different types of plots into a figure. Figure 1.10(e) shows a composite example made of a waterfall plot for the spectra and three-dimensional contour plots to identify key physics. Here, colors and arrows are used to annotate the plots. Figures can also be compiled

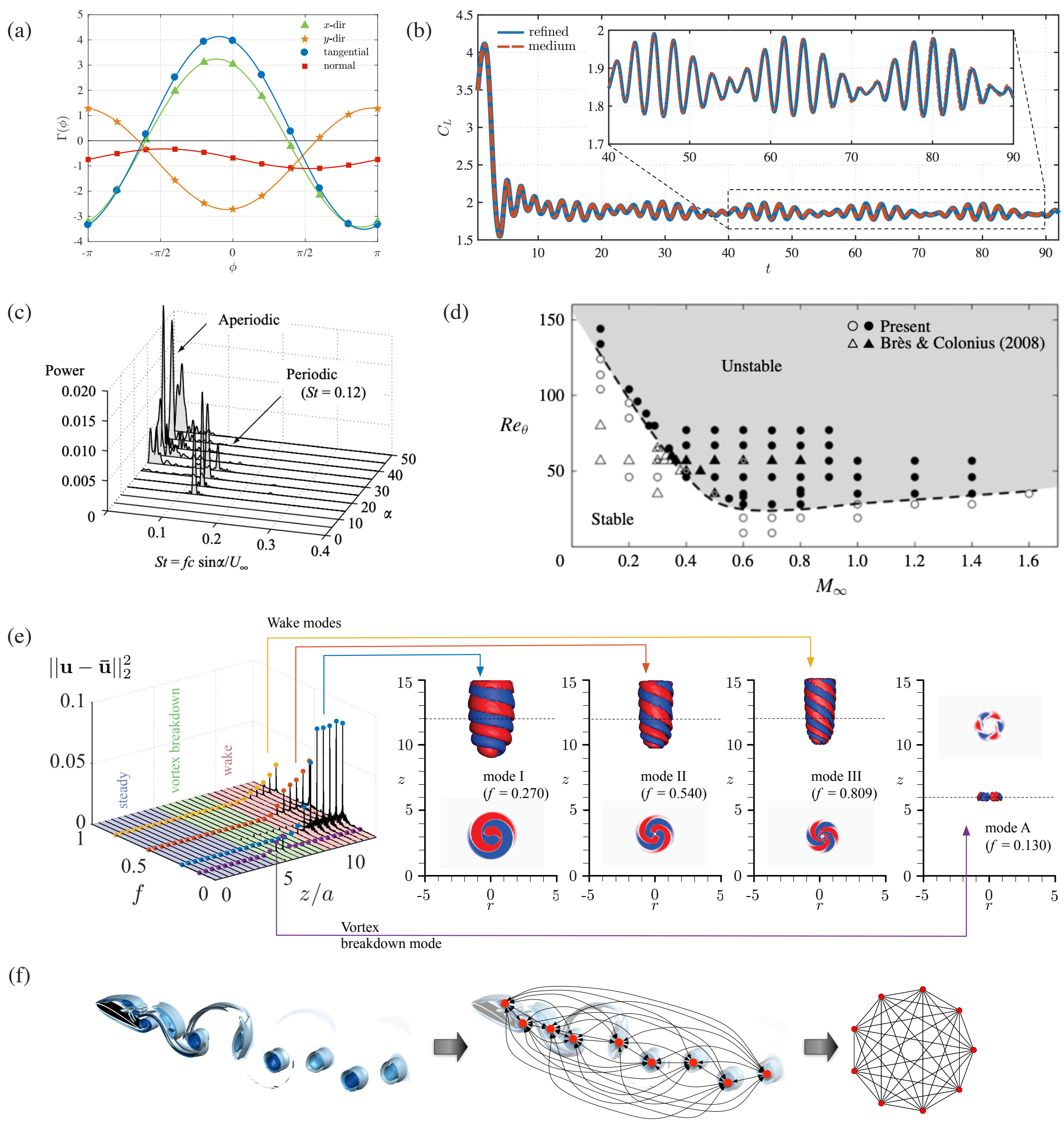

Figure 1.10: A gallery of plots. (a) Curves distinguished by color and symbols [4]. (b) Two curves with a zoomed inset [5]. (c) Stacked spectra showing transition [6] (d) Use of shading and symbols to demarcate regions [7]. (e) A compilation of spectra on the left with corresponding three-dimensional plots on the right [8]. (f) Transition from physics to mathematical abstraction.

to show transitions of ideas. In figure 1.10(f), the conceptional transition from a physical phenomenon to a mathematical abstraction is presented by using transparency and arrows. This type of plot can be very powerful in presentations and proposals.

With any of these plots, one important point to remember is not to over-complicate the graphs. Plots should be helpful to the audience but not confusing. A rule of thumb is to ensure your plots can be understood fairly well without having to read their captions or the main text. However, details of the plot should be explained thoroughly in the text to provide further insights.

1.6 Overview diagrams

I personally like the use of an overview diagram at the beginning of a paper to provide a visual summary of the paper. For papers that are fairly easy to follow, this type of plot may not be needed. However, these overview plots can be very valuable for extended papers. Shown in figure 1.11 is an overview plot from a 39-page paper [9]. By visually presenting the roadmap of the work contained in the extended paper, the authors can provide guidance for the audience on what lies ahead. It can also motivate the readers to continue reading your paper.

This type of figure is also immensely useful in presentations to communicate the key topics in a swift and motivating manner[11]. If done right, this type of plot can be shared easily between the paper and the presentation slides. Generally, nice plots can be shared across paper and presentations. A good plot goes a long way in many ways.

[11] In our field of research, we give 10-minute talks at the APS DFD Meetings. Overview diagrams are wonderful for such short talks

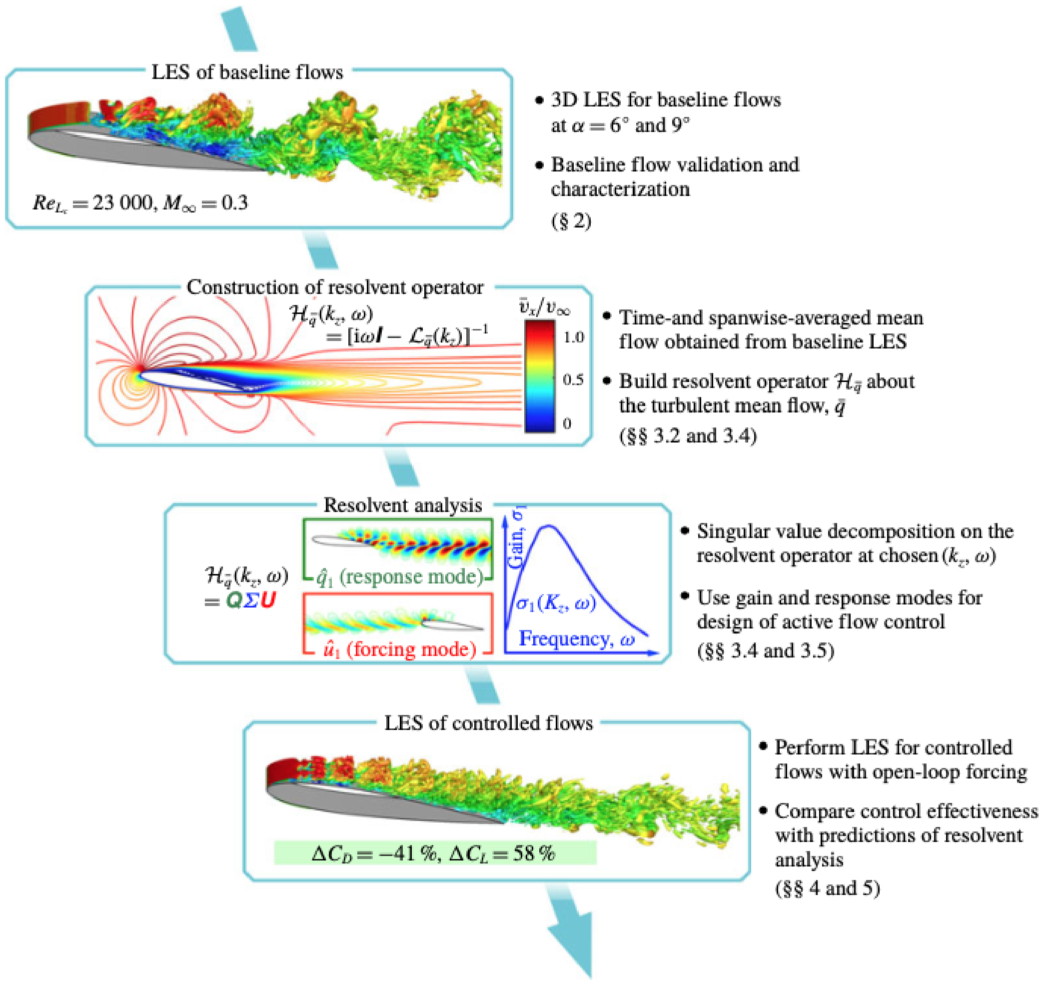

Figure 1.11: An example overview plot [9].

Bibliography

[1] J. Atkins, "othercolor." Available at https://www.mathworks.com/matlabcentral/fileexchange/30564-othercolor.

[2] National Institute of Health, "Color vision deficiency." Available at https://ghr.nlm.nih.gov/condition/color-vision-deficiency#statistics.

[3] V. Atanasiu, "Matlab colormaps as seen by color-blind users." Available at https://www.mathworks.com/matlabcentral/fileexchange/30161-matlab-colormaps-as-seen-by-color-blind-users.

[4] K. Taira and H. Nakao, "Phase-response analysis of synchronization for periodic flows," *J. Fluid Mech.*, vol. 846, p. R2, 2018.

[5] K. Zhang, S. Hayostek, M. Amitay, W. He, V. Theofilis, and K. Taira, "On the formation of three-dimensional separated flows over wings under tip effects," *J. Fluid Mech.*, vol. 895, p. A9, 2020.

[6] K. Taira and T. Colonius, "Three-dimensional flows around low-aspect-ratio flat-plate wings at low Reynolds numbers," *J. Fluid Mech.*, vol. 623, pp. 187–207, 2009.

[7] Y. Sun, K. Taira, L. N. Cattafesta, and L. S. Ukeiley, "Biglobal stability analysis of compressible open cavity flow," *J. Fluid Mech.*, vol. 826, pp. 270–301, 2017.

[8] Q. Liu, B. An, M. Nohmi, M. Obuchi, and K. Taira, "Core-pressure alleviation for a wall-normal vortex by active flow control," *J. Fluid Mech.*, vol. 853, p. R1, 2018.

[9] C.-A. Yeh and K. Taira, "Resolvent-analysis-based design of airfoil separation control," *J. Fluid Mech.*, vol. 867, pp. 572–610, 2019.