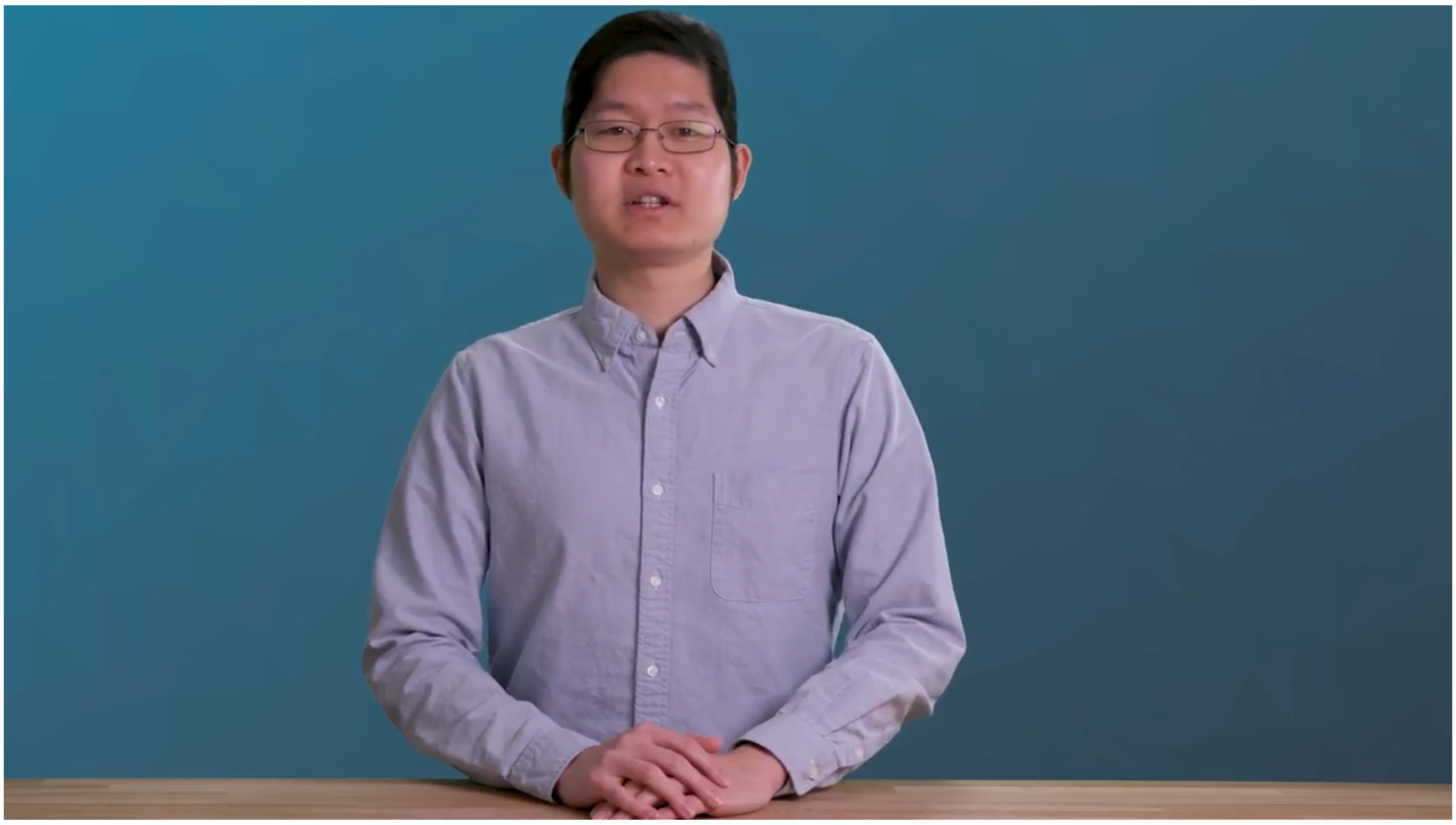
# 06. Polishing Plots

## L6 061 Polishing Plots V3



---

## Polishing Plots

Thus far in the course, the code you've seen has been fairly bare-bones, just enough to get your plots created. In order to convey your findings to others quickly and efficiently, you'll need to put work into polishing your plots. There are many dimensions to consider when putting together a polished plot.

### Choose an appropriate plot

Your choice of plot will depend on the number of variables that you have and their types: nominal, ordinal, discrete numeric, or continuous. Choice of plot also depends on the specific relationship that you want to convey. For example, whether you choose a violin plot, box plot, or adapted bar chart depends on how much data you have and whether distributions are significant or important. You'll be more likely to use a violin plot if you have a lot of data and the distributions are meaningful, and more inclined to use a box plot or bar chart if you have less data, or the distributions are less reliable.

### Choose appropriate encodings

Your variables should impact not just the type of plot that is chosen, but also the variable encodings. For example, if you have three numeric variables, you shouldn't just assign x-position, y-position, and color encodings randomly. In many cases, the two variables that are most important should take the positional encodings; if one represents an outcome or dependent variable, then it should be plotted on the y-axis. In other cases, it makes sense to plot the dependent measure with color, as though you are taking a top-down view of the plane defined by the two independent measures plotted on the axes.

### Pay attention to design integrity

When setting up your plotting parameters, remember the design principles from earlier in the course.

Make sure that you minimize chart junk and maximize the data-ink ratio, as far as it maintains good interpretability of the data. When deciding whether or not to add non-positional encodings, make sure that they are meaningful. For example, using color in a frequency bar chart may not be necessary on its own, but will be useful if those colors are used again later in the same presentation, matched with their original groups. By the same token, avoid using the same color scheme for different variables to minimize the chance of reader confusion.

You should also ensure that your plot avoids lie factors as much as possible. If you have a bar chart or histogram, it is best to anchor them to a 0 baseline. If you're employing a scale transformation, signal this clearly in the title, axis labels, and tick marks.

### Label axes and choose appropriate tick marks

For your positional axes, make sure you include axis labels. This is less important in exploration when you have the code available and have an extended flow to your work, but when you're conveying only the key pieces to others, it's crucial. When you add an axis label, make sure you also provide the units of measurement, if applicable (e.g., stating "Height (cm)" rather than just "Height").

As for tick marks, you should include at least three tick marks on each axis. This is especially important for data that has been transformed: you want enough tick marks so that the scale of the data can be communicated there. If your values are very large or very small numbers, consider using abbreviations to relabel the ticks (e.g., use "250K" instead of "250000").

### Provide legends for non-positional variables

Make sure that you add a legend for variables not depicted on the axes of your plot. For color encoding, you can add a color bar to the side of the plot. The most important new thing here is that you provide a descriptive label to your legend or color bar, just as you would the axes of your plot.

### Title your plot and include descriptive comments

Finally, make sure that you provide a descriptive title to your plot. If this is a key plot that presents important findings to others, aim to create a title that draws attention to those main points, rather than just state what variables are plotted.

Also, realize that while a visualization might be the core mechanism by which you convey findings, it need not stand alone. Comments in the text below or surrounding the plot can provide valuable context to help the reader understand your message, or reinforce the main points that they should have gotten.

---

## Using Matplotlib to Polish Plots

Back in the univariate plots lesson, you were introduced to the general way that visualizations are structured in matplotlib and seaborn: each visualization is based off a single Figure, which contains one or more Axes, and each Axes houses elements like points, lines, and boxes that depict the plotted data. Understanding and making use of this structure will open up your ability to polish your visualizations. Each function below is linked to its documentation page and which object type it is associated with.

- `figure` (Figure): Used to create a new figure. You'll use this first to initialize the figure, most often using the "figsize" parameter to set the figure dimensions.
- `xlabel` and `ylabel` (Axes): Used for setting axis labels.
- `xticks` and `yticks` (Axes): Used for setting tick marks.
- `legend` (Axes): Used to create and customize a legend. One key parameter to use is "title", which allows you to label what feature is being depicted in the legend. You might also need to make use of the "loc" and "ncol" parameters to move and shape the legend if it gets placed in an awkward location by default.
- `colorbar` (Axes): Used to add a colorbar to a plot. Use the "label" parameter to set the label on a colorbar.
- `title` (Axes): Used for setting axis titles.
- `suptitle` (Figure): Used for setting figure titles. The main difference between `suptitle` and `title` is that the former sets a title for the Figure as a whole, and the latter for only a single Axes. This is an important distinction: if you're using faceting or subplotting, you'll want to use `suptitle` to set a title for the figure as a whole.

All of the functions above, or parameters associated with those functions in the case of seaborn, have been used sporadically throughout the course. Below are a couple examples of these polishing functions in use.

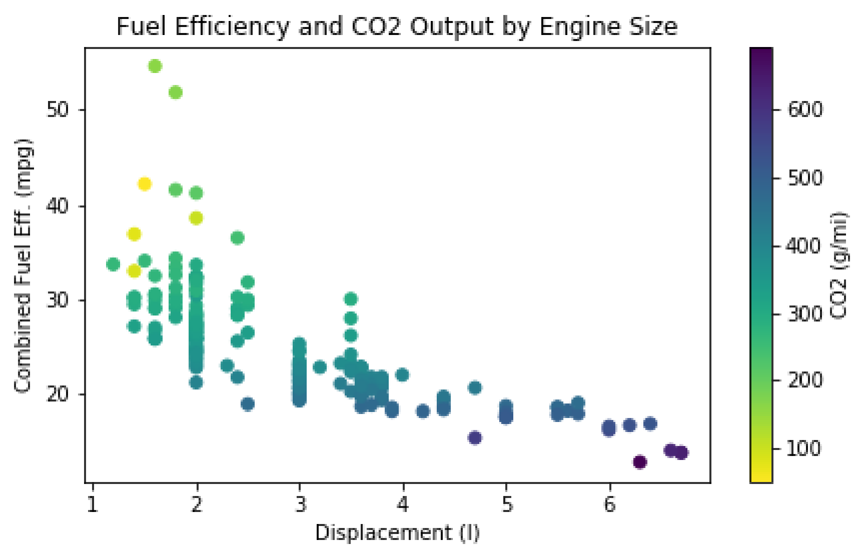
---

This example makes use of the fuel economy dataset. Since a colorbar is being added to the plot, `figsize` is used to make the figure a little bit wider than normal. Title, axes, and colorbar are all labeled. Note how the units of each feature being plotted is given in parentheses in each label.

```
# loading in the data, sampling to reduce points plotted
fuel_econ = pd.read_csv('./data/fuel_econ.csv')

np.random.seed(2018)
sample = np.random.choice(fuel_econ.shape[0], 200, replace = False)
fuel_econ_subset = fuel_econ.loc[sample]

# plotting the data
plt.figure(figsize = [7,4])
plt.scatter(data = fuel_econ_subset, x = 'displ', y = 'comb', c = 'co2',
            cmap = 'viridis_r')
plt.title('Fuel Efficiency and CO2 Output by Engine Size')
plt.xlabel('Displacement (l)')
plt.ylabel('Combined Fuel Eff. (mpg)')
plt.colorbar(label = 'CO2 (g/mi)');
```



Next Concept