**Seaborn Styling, Part 2: Color**

Learn how to work with color in Seaborn and choose appropriate color palettes for your datasets.

### Introduction

When creating a data visualization, your goal is to communicate the insights found in the data. While visualizing communicates important information, styling will influence how your audience understands what you're trying to convey.

After you have formatted and visualized your data, the third and last step of data visualization is styling. Styling is the process of customizing the overall look of your visualization, or figure. Making intentional decisions about the details of the visualization will increase their impact and set your work apart.

In this article, we'll look at how we can effectively use color to convey meaning. We'll cover:

* How to set a palette
* Seaborn default and built-in color palettes
* Color Brewer Palettes
* Selecting palettes for your dataset

#### Commands for Working with Palettes

You can build color palettes using the function sns.color\_palette(). This function can take any of the Seaborn built-in palettes (see below). You can also build your own palettes by passing in a list of colors in any valid Matplotlib format, including RGB tuples, hex color codes, or HTML color names.

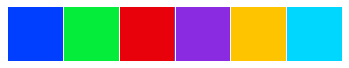
If you want to quickly see what a palette looks like, use the function sns.palplot() to plot a palette as an array of colors:

# Save a palette to a variable:

palette = sns.color\_palette("bright")

# Use palplot and pass in the variable:

sns.palplot(palette)



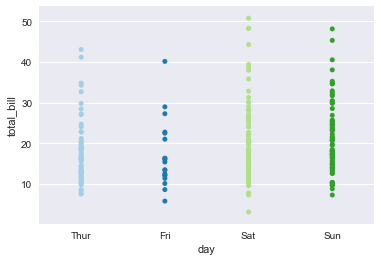
To select and set a palette in Seaborn, use the command sns.set\_palette() and pass in the name of the palette that you would like to use:

# Set the palette using the name of a palette:

sns.set\_palette("Paired")

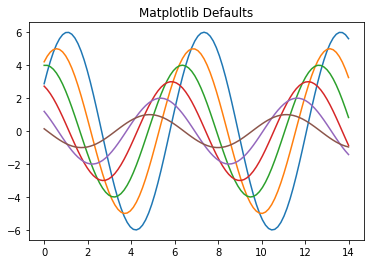
# Plot a chart:

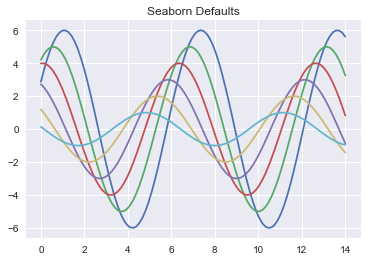
sns.stripplot(x="day", y="total\_bill", data=tips)



#### Seaborn Default Color Palette

If you do not pass in a color palette to sns.color\_palette() or sns.set\_palette(), Seaborn will use a default set of colors. These defaults improve upon the Matplotlib default color palettes and are one significant reason why people choose to use Seaborn for their data visualizations. Here's a comparison of the two default palettes:





Seaborn also allows you to style Matplotlib plots. So even if you're using a plot that only exists in Matplotlib, such as a histogram, you can do so using Seaborn defaults.

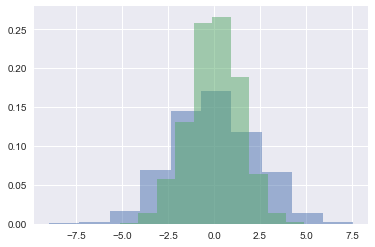
To do so, call the sns.set() function before your plot:

# Call the sns.set() function

sns.set()

for col in 'xy':

plt.hist(data[col], normed=True, alpha=0.5)



Not only does this function allow you the ability to use Seaborn default colors, but also any of Seaborn's other styling techniques.

Seaborn has six variations of its default color palette: deep, muted, pastel, bright, dark, and colorblind.



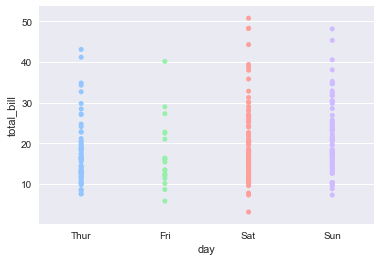
To use one of these palettes, pass the name into sns.set\_palette():

# Set the palette to the "pastel" default palette:

sns.set\_palette("pastel")

# plot using the "pastel" palette

sns.stripplot(x="day", y="total\_bill", data=tips)



### Using **Color Brewer Palettes**

In addition to the default palette and its variations, Seaborn also allows the use of Color Brewer palettes. [Color Brewer](http://colorbrewer2.org/) is the name of a set of color palettes inspired by the research of cartographer [Cindy Brewer](http://www.personal.psu.edu/cab38/). The color palettes are specifically chosen to be easy to interpret when used to represent ordered categories. They are also colorblind accessible, as each color differs from its neighbors in lightness or tint.

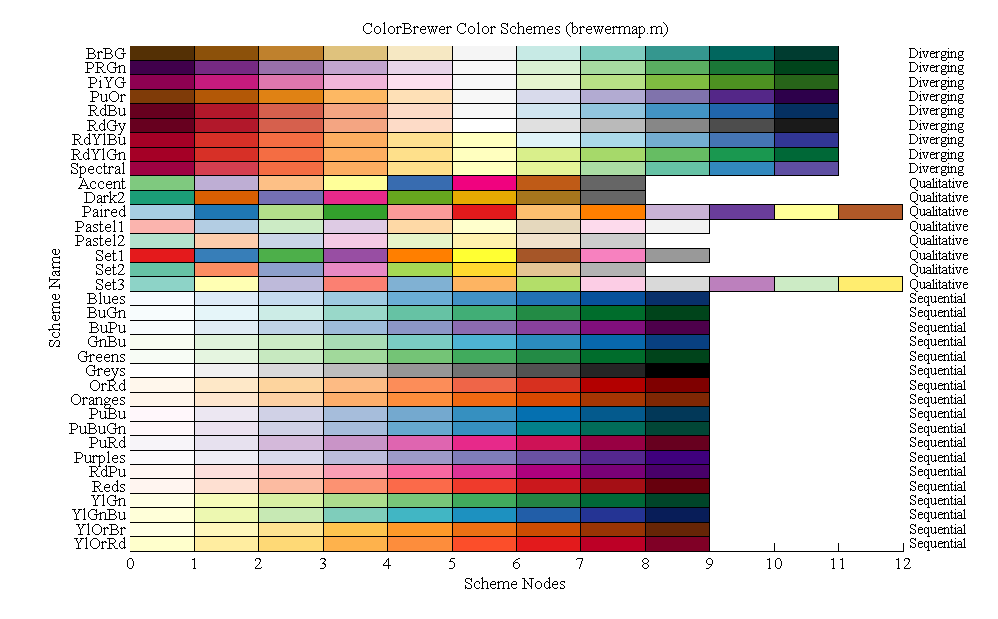
To use, pass the name of any Color Brewer palette directly to any of the color functions:

custom\_palette = sns.color\_palette("Paired", 9)

sns.palplot(custom\_palette)



Here is a list of the the Color Brewer palettes, with their names for easy reference:



Check out [http://colorbrewer2.org](http://colorbrewer2.org/#type=sequential&scheme=BuGn&n=3) for more information about color palette configuration options.

### Selecting Color Palettes for Your Dataset

#### Qualitative Palettes for Categorical Datasets

When using a dataset that uses distinct but non-ordered categories, it's good to use qualitative palettes. Qualitative palettes are sets of distinct colors which make it easy to distinguish the categories when plotted but don't imply any particular ordering or meaning.

An example of categorical data is breed of dog. Each of these values, such as Border Collie or Poodle, are distinct from each other but there isn't any inherent order to these categories.

Here's an example of a qualitative Color Brewer palette:

qualitative\_colors = sns.color\_palette("Set3", 10)

sns.palplot(qualitative\_colors)



#### Sequential Palettes

Just as the name implies, sequential palettes are a set of colors that move sequentially from a lighter to a darker color. Sequential color palettes are appropriate when a variable exists as ordered categories, such as grade in school, or as continuous values that can be put into groups, such as yearly income. Because the darkest colors will attract the most visual attention, sequential palettes are most useful when only high values need to be emphasized.

Here's an example of a sequential Color Brewer palette:

sequential\_colors = sns.color\_palette("RdPu", 10)

sns.palplot(sequential\_colors)



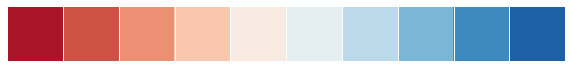
#### Diverging Palettes

Diverging palettes are best suited for datasets where both the low and high values might be of equal interest, such as hot and cold temperatures.

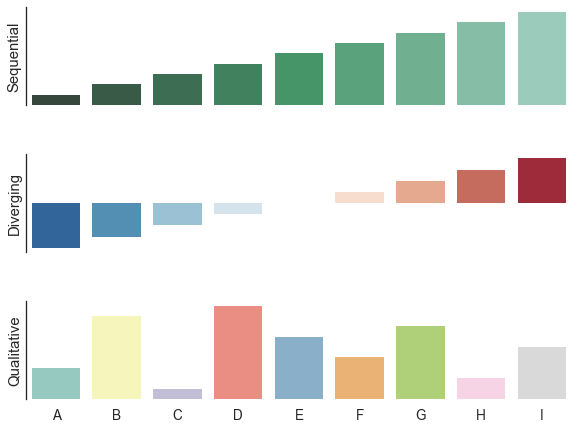
In the example below, both ends of the spectrum — fire red and deep blue — are likely to attract attention.

diverging\_colors = sns.color\_palette("RdBu", 10)

sns.palplot(diverging\_colors)



Here is a quick diagram that depicts each of the palette types:



###### [Credit: Michael Waskom](https://seaborn.pydata.org/examples/color_palettes.html)

### Summary

The ability to use easily choose different color palettes is one of the important affordances of styling your plots with Seaborn. Seaborn gives you a range of built-in plots to choose from: whether it's variations on the defaults or access to all of the Color Brewer palettes. It's easy to choose a palette that is well suited to your dataset, thanks to Color Brewer, as it supports palettes for qualitative, sequential, and diverging datasets.

For more on using color in Seaborn, check out their [documentation](https://seaborn.pydata.org/tutorial/color_palettes.html?highlight=color).