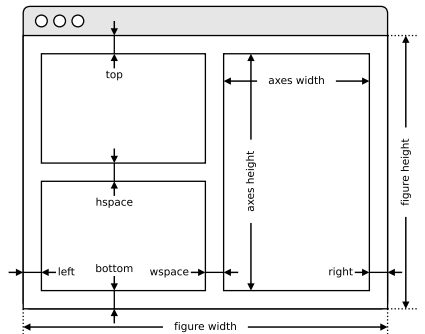




## Axes adjustments

API

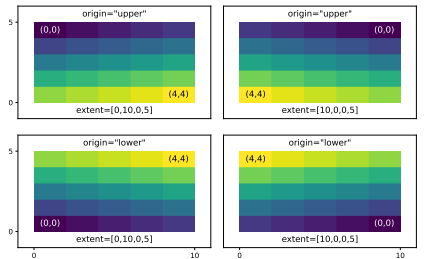
`plt.subplots_adjust(...)`



## Extent & origin

API

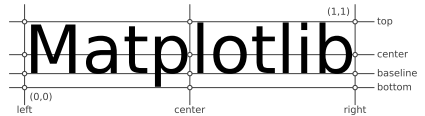
`ax.imshow(..., origin=...)`



## Text alignments

API

`ax.text(..., ha=..., va=..., ...)`



## Text parameters

API

`ax.text(..., family=..., size=..., weight=...)`  
`ax.text(..., fontproperties=...)`

The quick brown fox  
The quick brown fox  
The quick brown fox  
The quick brown fox  
The quick brown fox  
The quick brown fox  
The quick brown fox

xx-large (1.73)  
x-large (1.44)  
large (1.20)  
medium (1.00)  
small (0.83)  
x-small (0.69)  
xx-small (0.58)

**The quick brown fox jumps over the lazy dog**  
**The quick brown fox jumps over the lazy dog**  
**The quick brown fox jumps over the lazy dog**  
**The quick brown fox jumps over the lazy dog**  
**The quick brown fox jumps over the lazy dog**  
**The quick brown fox jumps over the lazy dog**  
**The quick brown fox jumps over the lazy dog**

black (900)  
bold (700)  
semibold (600)  
normal (400)  
ultralight (100)

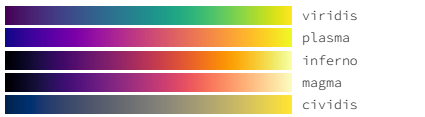
The quick brown fox jumps over the lazy dog  
The quick brown fox jumps over the lazy dog  
The quick brown fox jumps over the lazy dog  
*The quick brown fox jumps over the lazy dog*  
The quick brown fox jumps over the lazy dog  
The quick brown fox jumps over the lazy dog

monospace  
serif  
sans  
cursive  
italic  
normal

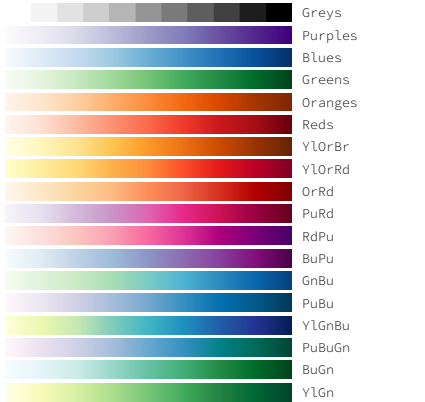
THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG  
The quick brown fox jumps over the lazy dog

small-caps  
normal

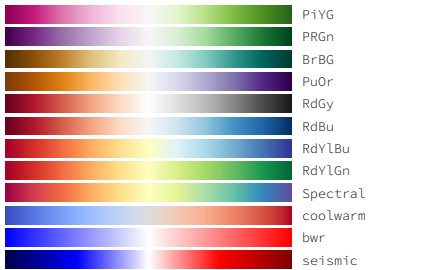
## Uniform colormaps



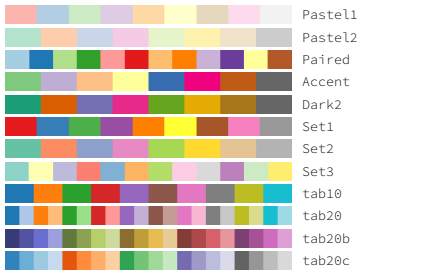
## Sequential colormaps



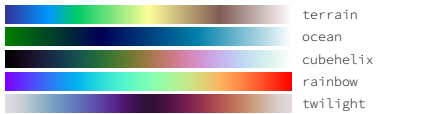
## Diverging colormaps



## Qualitative colormaps



## Miscellaneous colormaps

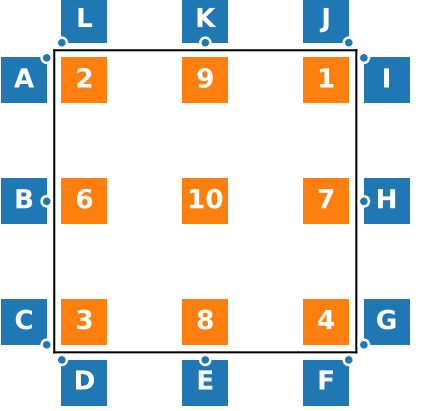


## Color names

API



## Legend placement



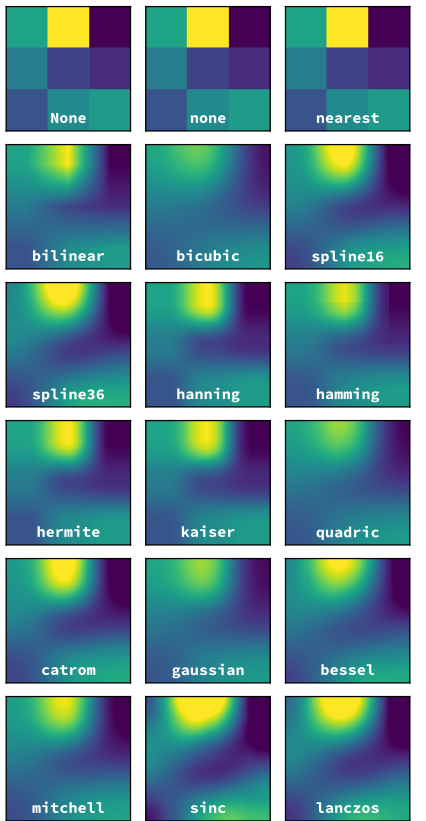
`ax.legend(loc="string", bbox_to_anchor=(x,y))`

2: upper left      9: upper center      1: upper right  
6: center left      10: center      7: center right  
3: lower left      8: lower center      4: lower right

A: upper right / (-0.1, 0.9)      B: center right / (-0.1, 0.5)  
C: lower right / (-0.1, 0.1)      D: upper left / (0.1, -0.1)  
E: upper center / (0.5, -0.1)      F: upper right / (0.9, -0.1)  
G: lower left / (1.1, 0.1)      H: center left / (1.1, 0.5)  
I: upper left / (1.1, 0.9)      J: lower right / (0.9, 1.1)  
K: lower center / (0.5, 1.1)      L: lower left / (0.1, 1.1)

## Image interpolation

API



## How do I ...

... **resize a figure?**  
→ `fig.set_size_inches(w, h)`

... **save a figure?**  
→ `fig.savefig("figure.pdf")`

... **save a transparent figure?**  
→ `fig.savefig("figure.pdf", transparent=True)`

... **clear a figure/an axes?**  
→ `fig.clear()` → `ax.clear()`

... **close all figures?**  
→ `plt.close("all")`

... **remove ticks?**  
→ `ax.set_[xy]ticks([])`

... **remove tick labels?**  
→ `ax.set_[xy]ticklabels([])`

... **rotate tick labels?**  
→ `ax.tick_params(axis="x", rotation=90)`

... **hide top spine?**  
→ `ax.spines['top'].set_visible(False)`

... **hide legend border?**  
→ `ax.legend(frameon=False)`

... **show error as shaded region?**  
→ `ax.fill_between(X, Y+error, Y-error)`

... **draw a rectangle?**  
→ `ax.add_patch(plt.Rectangle((0, 0), 1, 1))`

... **draw a vertical line?**  
→ `ax.axvline(x=0.5)`

... **draw outside frame?**  
→ `ax.plot(..., clip_on=False)`

... **use transparency?**  
→ `ax.plot(..., alpha=0.25)`

... **convert an RGB image into a gray image?**  
→ `gray = 0.2989*R + 0.5870*G + 0.1140*B`

... **set figure background color?**  
→ `fig.patch.set_facecolor("grey")`

... **get a reversed colormap?**  
→ `plt.get_cmap("viridis_r")`

... **get a discrete colormap?**  
→ `plt.get_cmap("viridis", 10)`

... **show a figure for one second?**  
→ `fig.show(block=False, time.sleep(1))`

## Performance tips

`scatter(X, Y)`      slow  
`plot(X, Y, marker="o", ls="")`      fast

`for i in range(n): plot(X[i])`      slow  
`plot(sum([x+[None] for x in X], []))`      fast

`cla(), imshow(...), canvas.draw()`      slow  
`im.set_data(...), canvas.draw()`      fast

## Beyond Matplotlib

Seaborn: Statistical data visualization  
Cartopy: Geospatial data processing  
yt: Volumetric data visualization  
mpld3: Bringing Matplotlib to the browser  
Datashader: Large data processing pipeline  
plotnine: A grammar of graphics for Python

Matplotlib Cheatsheets  
Copyright (c) 2021 Matplotlib Development Team  
Released under a CC-BY 4.0 International License

NUMFOCUS  
OPEN CODE = BETTER SCIENCE