

Quick start

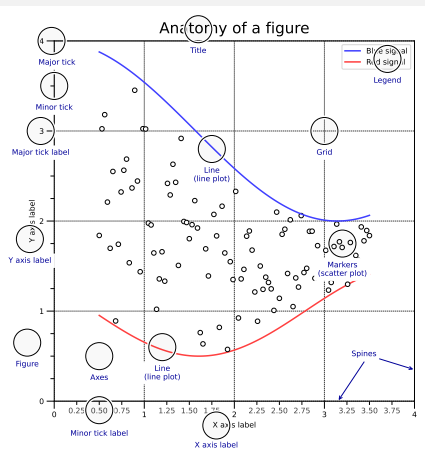
```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
```

```
X = np.linspace(0, 2*np.pi, 100)
Y = np.cos(X)
```

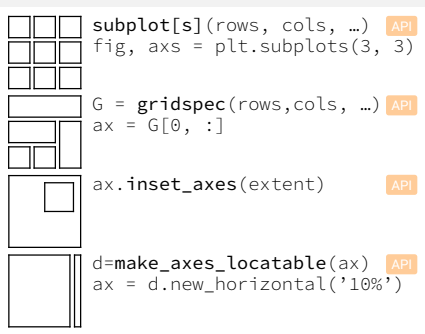
```
fig, ax = plt.subplots()
ax.plot(X, Y, color='green')
```

```
fig.savefig("figure.pdf")
plt.show()
```

Anatomy of a figure



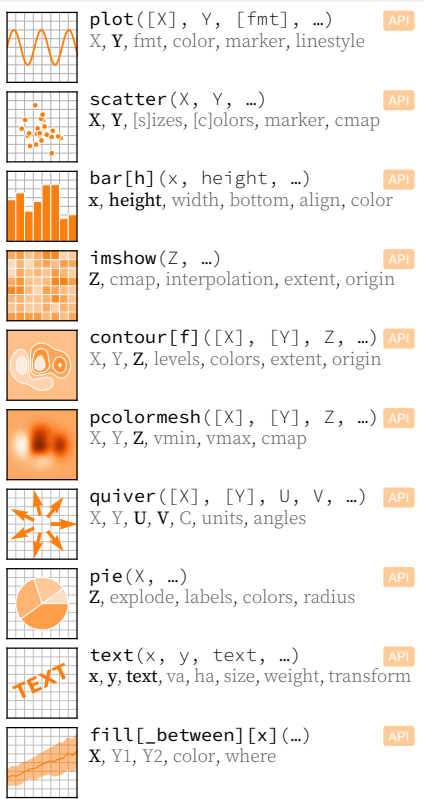
Subplots layout



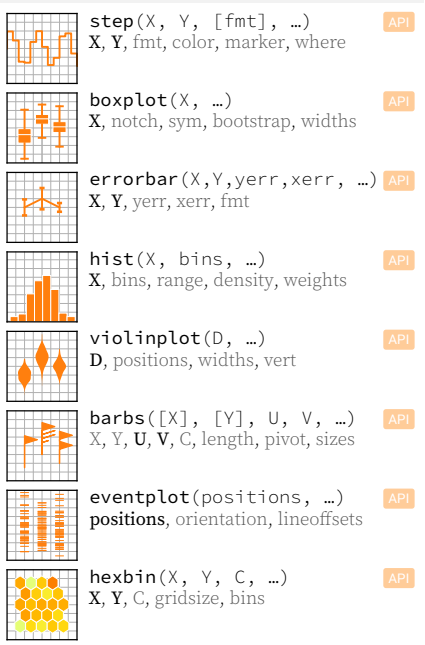
Getting help

matplotlib.org
discourse.matplotlib.org
stackoverflow.com/questions/tagged/matplotlib

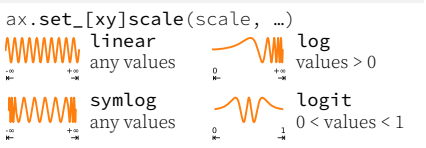
Basic plots



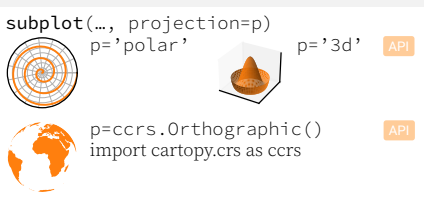
Advanced plots



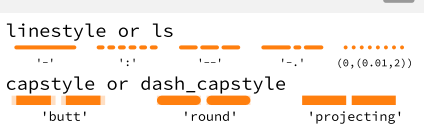
Scales



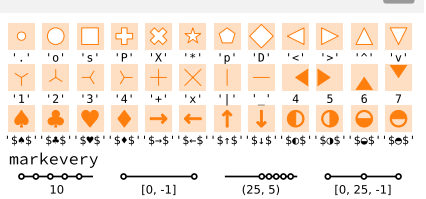
Projections



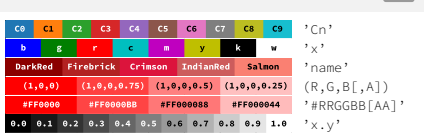
Lines



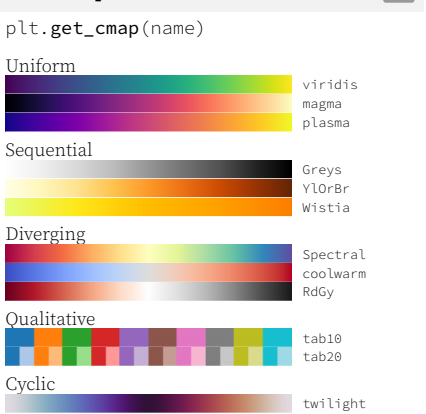
Markers



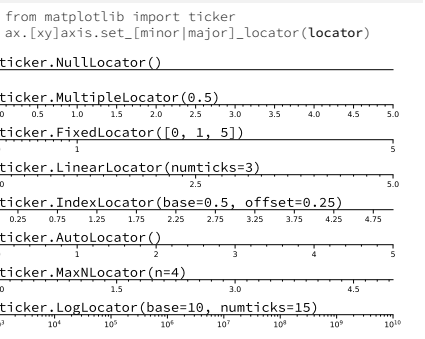
Colors



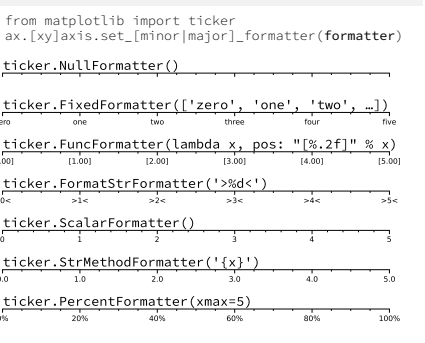
Colormaps



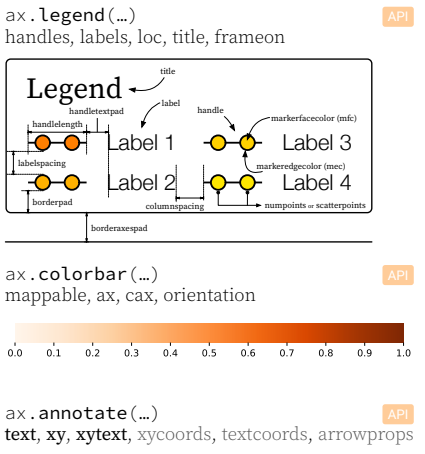
Tick locators



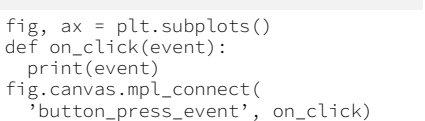
Tick formatters



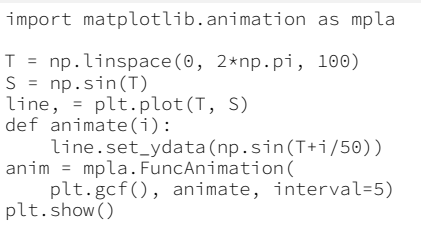
Ornaments



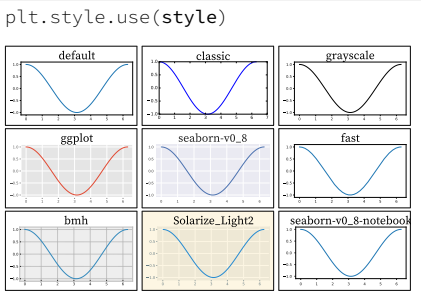
Event handling



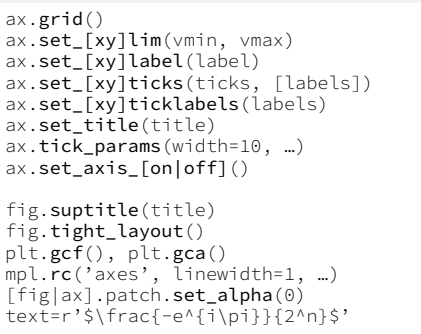
Animation



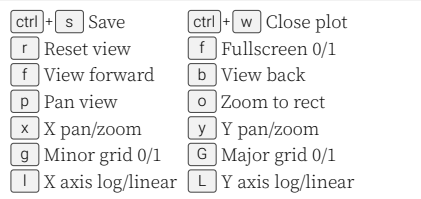
Styles



Quick reminder



Keyboard shortcuts

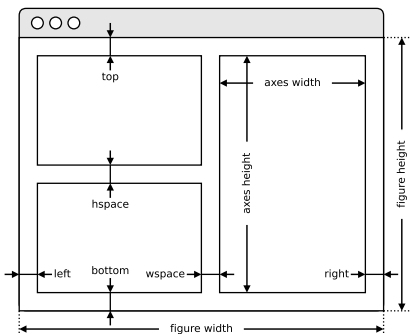


Ten simple rules

1. Know your audience
2. Identify your message
3. Adapt the figure
4. Captions are not optional
5. Do not trust the defaults
6. Use color effectively
7. Do not mislead the reader
8. Avoid "chartjunk"
9. Message trumps beauty
10. Get the right tool

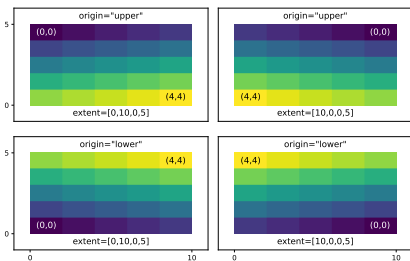
API

```
plt.subplots_adjust( ... )
```



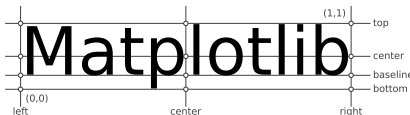
API

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



API

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



API

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

`ax.text(..., fontproperties=...)`

The quick brown fox	xx-large (1.73)
The quick brown fox	x-large (1.44)
The quick brown fox	large (1.20)
The quick brown fox	medium (1.00)
The quick brown fox	small (0.83)
The quick brown fox	x-small (0.69)
The quick brown fox	xx-small (0.58)

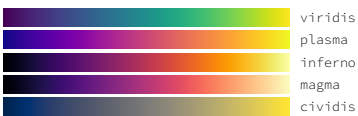
The quick brown fox jumps over the lazy dog	black (900)
The quick brown fox jumps over the lazy dog	bold (700)
The quick brown fox jumps over the lazy dog	semibold (600)
The quick brown fox jumps over the lazy dog	normal (400)
The quick brown fox jumps over the lazy dog	ultralight (100)

The quick brown fox jumps over the lazy dog monospace
The quick brown fox jumps over the lazy dog serif
The quick brown fox jumps over the lazy dog sans
The quick brown fox jumps over the lazy dog cursive

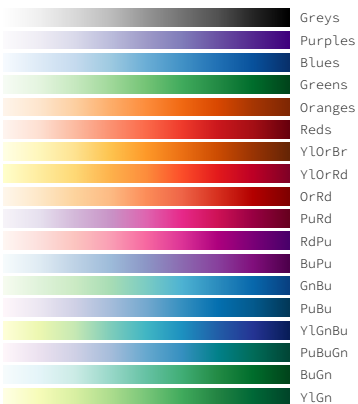
The quick brown fox jumps over the lazy dog italic

The quick brown fox jumps over the lazy dog normal

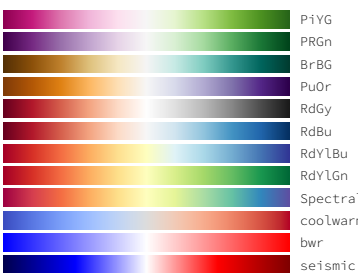
Uniform colormaps



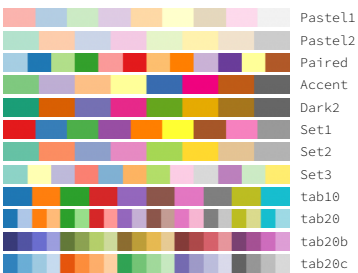
Sequential colormaps



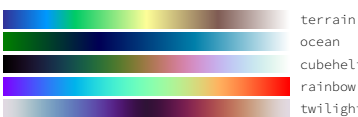
Diverging colormaps



Qualitative colormaps



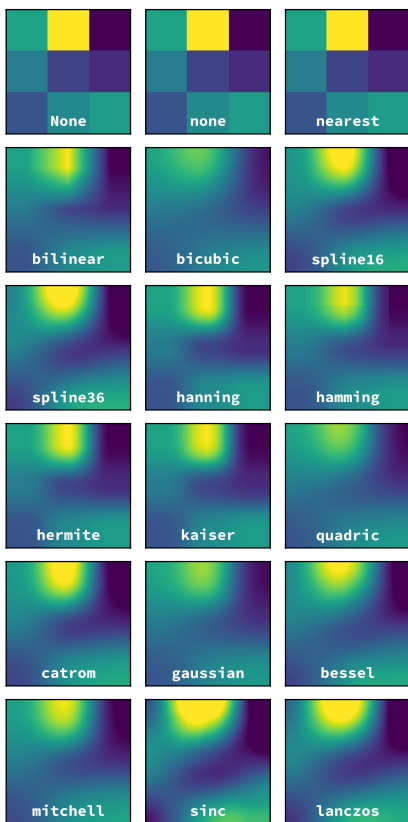
Miscellaneous colormaps



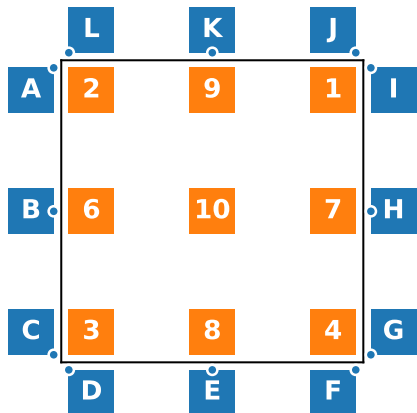
API



API



Legend placement

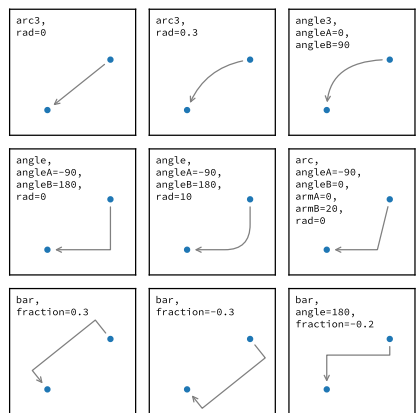


```
ax.legend(loc="string", bbox_to_anchor=(x, y))
```

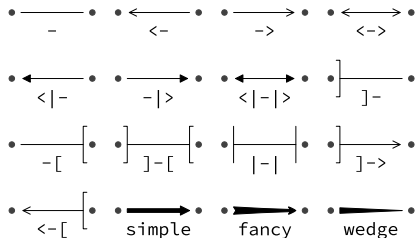
2: upper left 9: upper center 1: upper right
6: center left 10: center 7: center right
3: lower left 8: lower center 4: lower right

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 / $(0.1, 1.1)$

API



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(i, X[i], "o")	slow
plot(X, marker="o", ls="")	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
 Datastader: 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