matplotlib Cheat sheet

Version 3.5.0

Início rápido

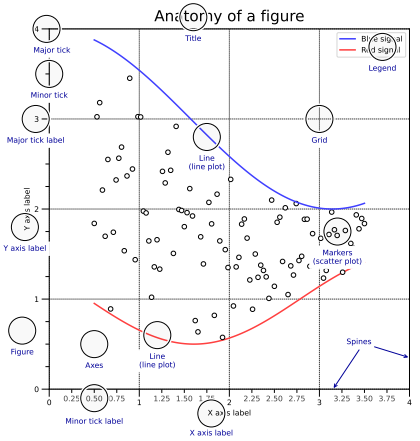
```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt


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")
fig.show()
```

Anatomia de uma figure



Layout de subplots

```
subplot[s](rows,cols,…)
fig, axs = plt.subplots(3, 3)

G = gridspec(rows,cols,…)
ax = G[0,:]

ax.inset_axes(extent)

d=make_axes_locatable(ax)
ax = d.new_horizontal('10%')
```

Conseguindo ajuda

- matplotlib.org
- github.com/matplotlib/matplotlib/issues
- discourse.matplotlib.org
- stackoverflow.com/questions/tagged/matplotlib
- https://gitter.im/matplotlib/matplotlib
- twitter.com/matplotlib
- Matplotlib users mailing list

Plots básicos

`plot([X],Y,[fmt],…)` API
X, Y, fmt, color, marker, linestyle

`scatter(X,Y,…)` API
X, Y, [s]izes, [c]olors, marker, cmap

`bar[h](x,height,…)` API
x, height, width, bottom, align, color

`imshow(Z,…)` API
Z, cmap, interpolation, extent, origin

`contour[f]([X],[Y],Z,…)` API
X, Y, Z, levels, colors, extent, origin

`pcolormesh([X],[Y],Z,…)` API
X, Y, Z, vmin, vmax, cmap

`quiver([X],[Y],U,V,…)` API
X, Y, U, V, C, units, angles

`pie(X,…)` API
Z, explode, labels, colors, radius

`text(x,y,text,…)` API
x, y, text, va, ha, size, weight, transform

`fill[_between][x](…)` API
X, Y1, Y2, color, where

Plots avançados

`step(X,Y,[fmt],…)` API
X, Y, fmt, color, marker, where

`boxplot(X,…)` API
X, notch, sym, bootstrap, widths

`errorbar(X,Y,xerr,yerr,…)` API
X, Y, xerr, yerr, fmt

`hist(X, bins, …)` API
X, bins, range, density, weights

`violinplot(D,…)` API
D, positions, widths, vert

`barbs([X],[Y], U, V, …)` API
X, Y, U, V, C, length, pivot, sizes

`eventplot(positions,…)` API
positions, orientation, lineoffsets

`hexbin(X,Y,C,…)` API
X, Y, C, gridsize, bins

Escalas

API

```
ax.set_[xy]scale(scale,…)
```

linear
any values

log
values > 0

symlog
any values

logit
0 < values < 1

Projeções

API

`subplot(…,projection=p)`

p='polar'

p='3d'

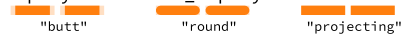
p=ccrs.Orthographic()
import cartopy.crs as ccrs

Linhas

API

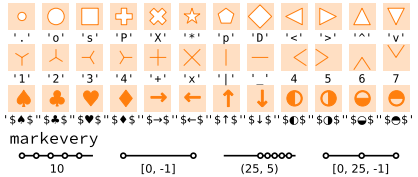
linestyle or ls

capstyle or dash_capstyle



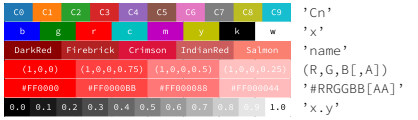
Markers

API



Cores

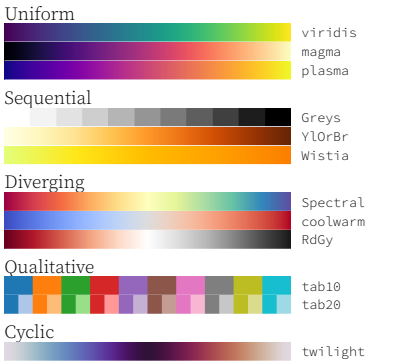
API



Mapas de cores

API

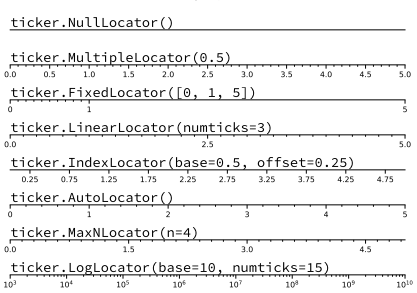
`plt.get_cmap(name)`



Localizações de ticks

API

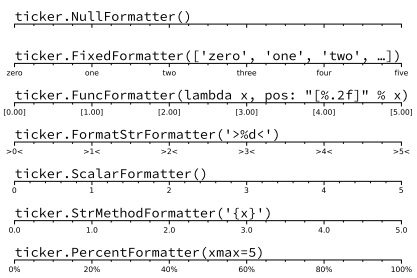
```
from matplotlib import ticker
ax.[xy]axis.set_[minor|major]_locator(locator)
```



Formatadores de ticks

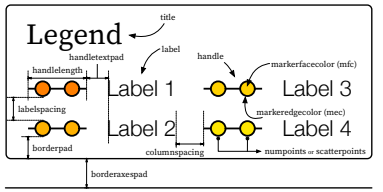
API

```
from matplotlib import ticker
ax.[xy]axis.set_[minor|major]_formatter(formatter)
```

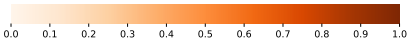


Ornamentos

`ax.legend(…)` API
handles, labels, loc, title, frameon



`ax.colorbar(…)` API
mappable, ax, cax, orientation



`ax.annotate(…)` API
text, **xy**, **xytext**, xycoords, textcoords, arrowprops

Event handling

API

```
fig, ax = plt.subplots()
def on_click(event):
  print(event)
fig.canvas.mpl_connect(
  'button_press_event', on_click)
```

Animação

API

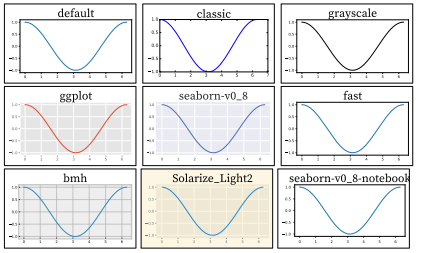
```
import matplotlib.animation as mpla

T = np.linspace(0, 2*np.pi, 100)
S = np.sin(T)
line, = plt.plot(T, S)
def animate(i):
    line.set_ydata(np.sin(T+i/50))
anim = mpla.FuncAnimation(
    plt.gcf(), animate, interval=5)
plt.show()
```

Estilos

API

`plt.style.use(style)`



Lembrete rápido

```
ax.grid()
ax.set_[xy]lim(vmin, vmax)
ax.set_[xy]label(label)
ax.set_[xy]ticks(ticks, [labels])
ax.set_[xy]ticklabels(labels)
ax.set_title(title)
ax.tick_params(width=10, …)
ax.set_axis_[on|off]()

fig.suptitle(title)
fig.tight_layout()
plt.gcf(), plt.gca()
mpl.rc('axes', linewidth=1, …)
[fig|ax].patch.set_alpha(0)
text=r'$\frac{-e^{i\pi}}{2^n}$'
```

Atalhos de teclado

API

ctrl + s	Save	ctrl + w	Close plot
r	Reset view	f	Fullscreen 0/1
f	View forward	b	View back
p	Pan view	o	Zoom to rect
x	X pan/zoom	y	Y pan/zoom
g	Minor grid 0/1	G	Major grid 0/1
l	X axis log/linear	L	Y axis log/linear

Dez regras simples

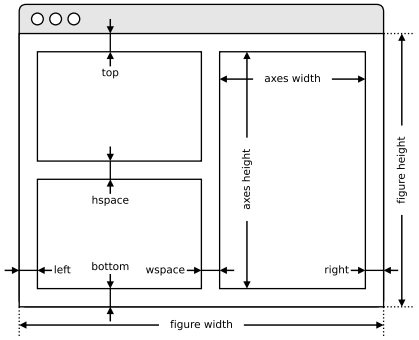
READ

1. Conheça sua audiência
2. Identifique sua mensagem
3. Adapte a figura
4. Legendas não são opcionais
5. Não confie nos defaults
6. Use cores efetivamente
7. Não confunda o leitor
8. Evite "chartjunk"
9. Messagem supera beleza
10. Tenha a ferramenta correta

Ajustes nos eixos

API

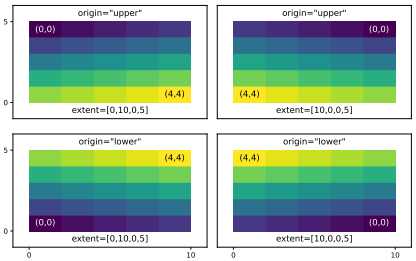
plt.**subplots_adjust**(...)



Extent & origin

API

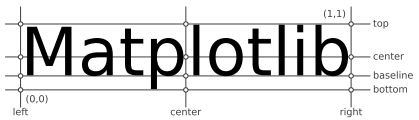
ax.**imshow**(extent=..., origin=...)



Alinhamento de texto

API

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



Parâmetros de texto

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)

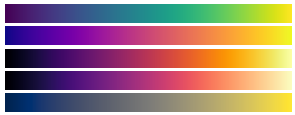
The quick brown fox jumps over the lazy dog — black (900)
The quick brown fox jumps over the lazy dog — bold (700)
The quick brown fox jumps over the lazy dog — semibold (600)
The quick brown fox jumps over the lazy dog — normal (400)
The quick brown fox jumps over the lazy dog — ultralight (100)

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

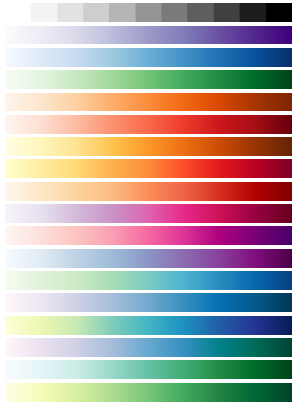
The quick brown fox jumps over the lazy dog — italic
The quick brown fox jumps over the lazy dog — normal

THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG — small-caps
The quick brown fox jumps over the lazy dog — normal

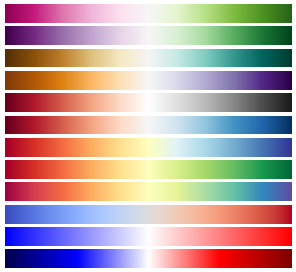
Mapas de cor uniforme



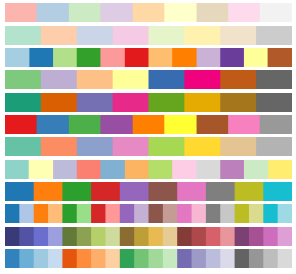
Mapas de cor sequenciais



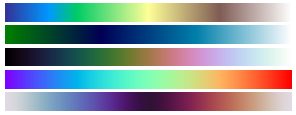
Mapas de cores divergentes



Mapas de cores qualitativos



Mapas de cores diversos

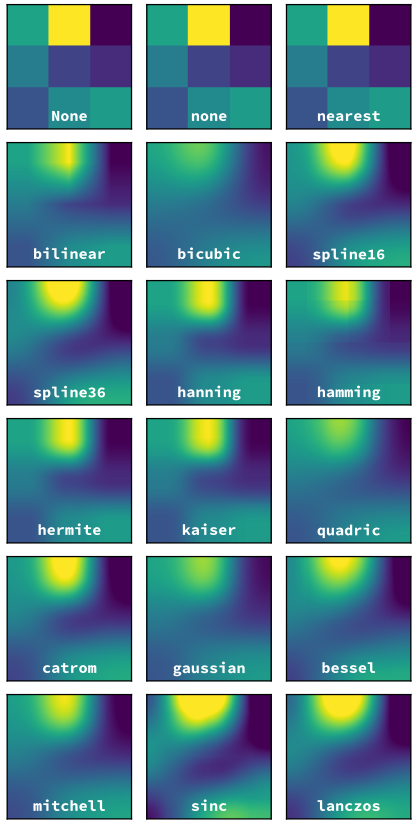


Nomes de cores

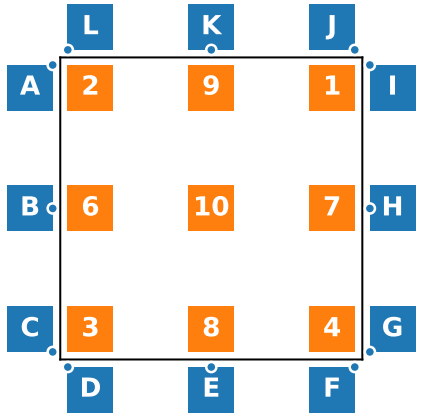
API

Interpolação de image

API



Localização da legenda



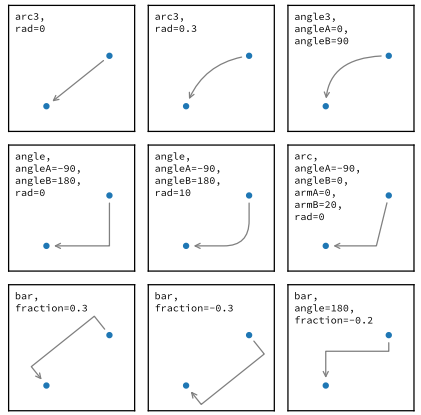
ax.**legend**(loc="string", bbox_to_anchor=(x,y))

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)`

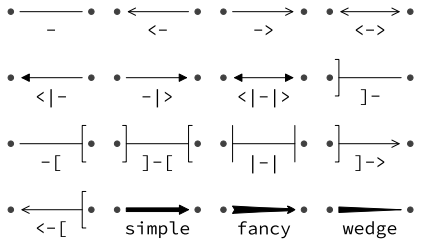
Estilo de anotação de conexões

API



Estilo de anotações de setas

API



Como eu ...

... redimensiono uma figura?
→ fig.set_size_inches(w, h)
... salvo uma figura?
→ fig.savefig("figure.pdf")
... salve uma figura transparente?
→ fig.savefig("figure.pdf", transparent=True)
... limpo uma figura/um eixo?
→ fig.clear() → ax.clear()
... fecho todas as figuras?
→ plt.close("all")
... removo ticks?
→ ax.set_[xy]ticks([])
... removo rótulos de ticks?
→ ax.set_[xy]ticklabels([])
... rotaciono rótulos de ticks?
→ ax.tick_params(axis="x", rotation=90)
... escondo top spine?
→ ax.spines['top'].set_visible(False)
... escondo borda da legenda?
→ ax.legend(frameon=False)
... mostro erro como uma região sombreada?
→ ax.fill_between(X, Y+error, Y-error)
... desenho um retângulo?
→ ax.add_patch(plt.Rectangle((0, 0), 1, 1)
... desenho uma linha vertical?
→ ax.axvline(x=0.5)
... desenho fora do frame?
→ ax.plot(..., clip_on=False)
... uso transparência?
→ ax.plot(..., alpha=0.25)
... converto uma imagem RGB em uma imagem cinza?
→ gray = 0.2989*R + 0.5870*G + 0.1140*B
... defino uma cor de fundo para a figura?
→ fig.patch.set_facecolor("grey")
... obtenho um mapa de cor reverso?
→ plt.get_cmap("viridis_r")
... obtenho um mapa de cor discreto?
→ plt.get_cmap("viridis", 10)
... mostro uma figura por um segundo?
→ fig.show(block=False), time.sleep(1)

Dicas de performance

```
scatter(X, Y)                          devagar
plot(X, Y, marker="o", ls="")          rápido

for i in range(n): plot(X[i])          devagar
plot(sum([x+[None] for x in X],[]))    rápido

cla(), imshow(…), canvas.draw()        devagar
im.set_data(…), canvas.draw()          rápido
```

Além do Matplotlib

Seaborn: Visualização de dado estatísticos
Cartopy: Processamento de dados geoespaciais
yt: Visualização de dados volumétricos
mpld3: Trazendo Matplotlib para o navegador
Datashader: Large data processing pipeline
plotnine: Uma gramática de gráficos para Python

NUMFOCUS
OPEN CODE = BETTER SCIENCE