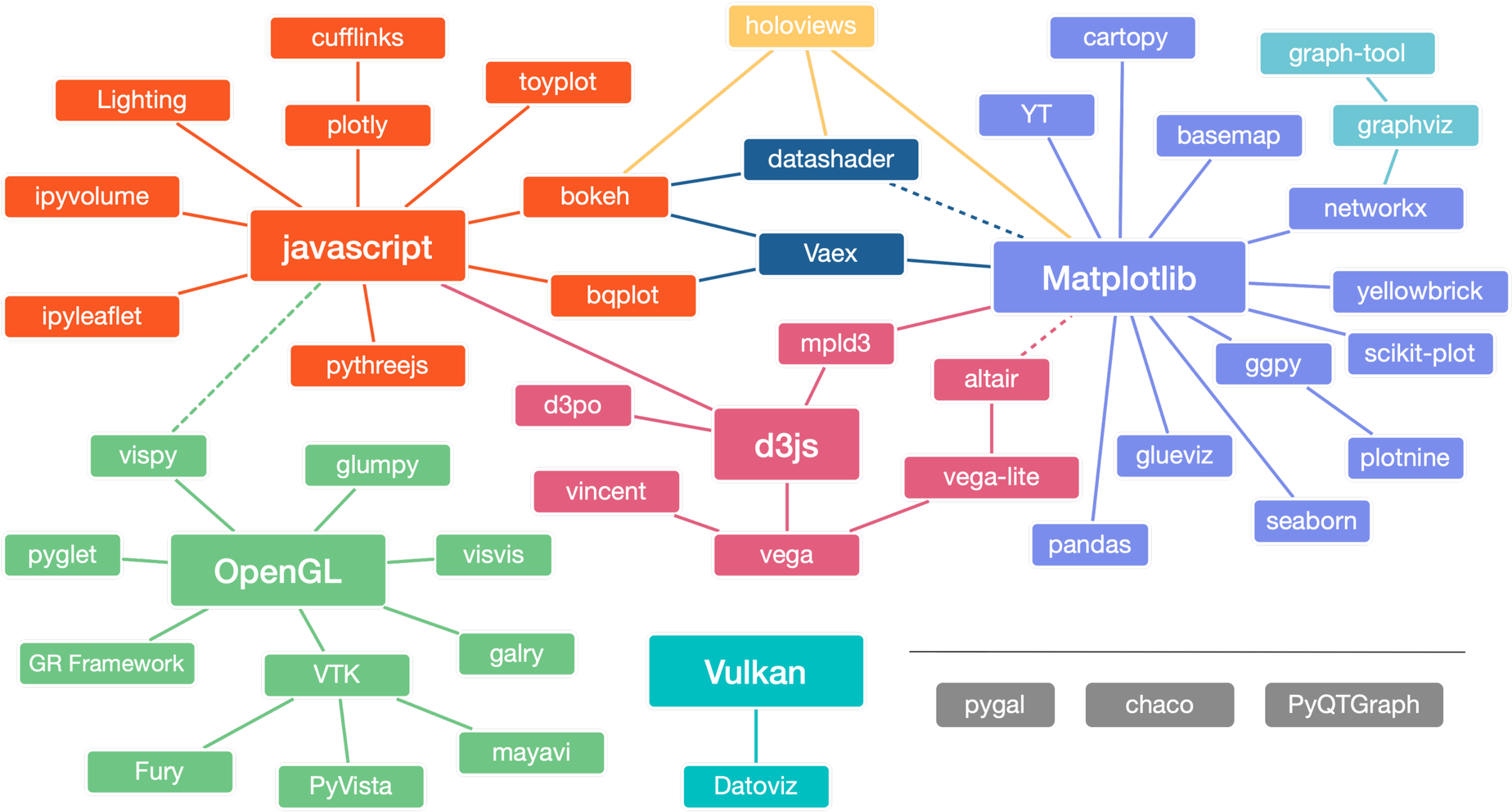
Data Science II

- Python Visualization Landscape -

Prof. Dr. Eduard Kromer
Summer Semester 2024
University of Applied Sciences Landshut

Python Visualization Landscape



How to choose?

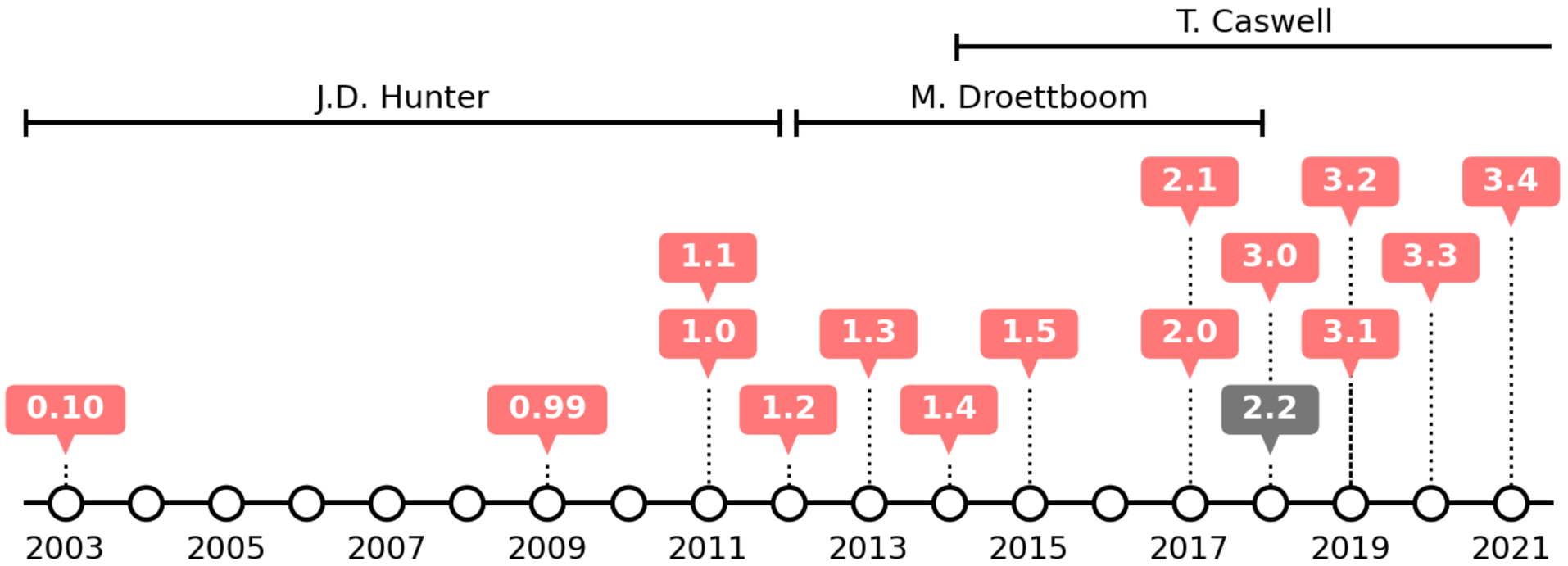
- desktop or web rendering (interactive visualization)?
- publication quality?
- data size?
- active community, good documentation, tutorials?

➡ visit https://pyviz.org and checkout the tutorials section

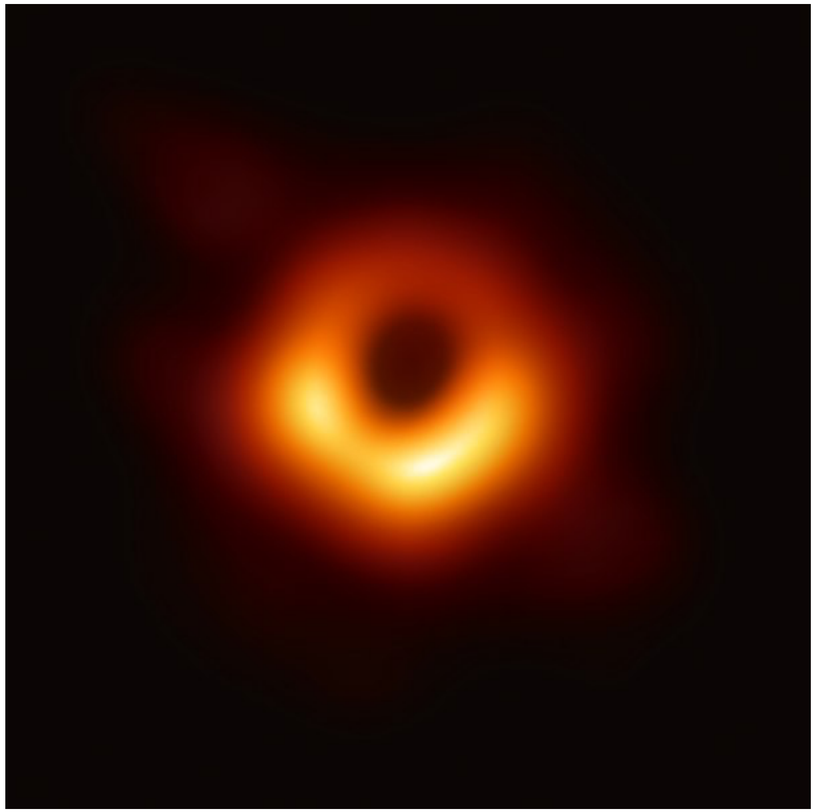
matplotlib

- the de facto standard for Python scientific visualization -

Matplotlib Timeline

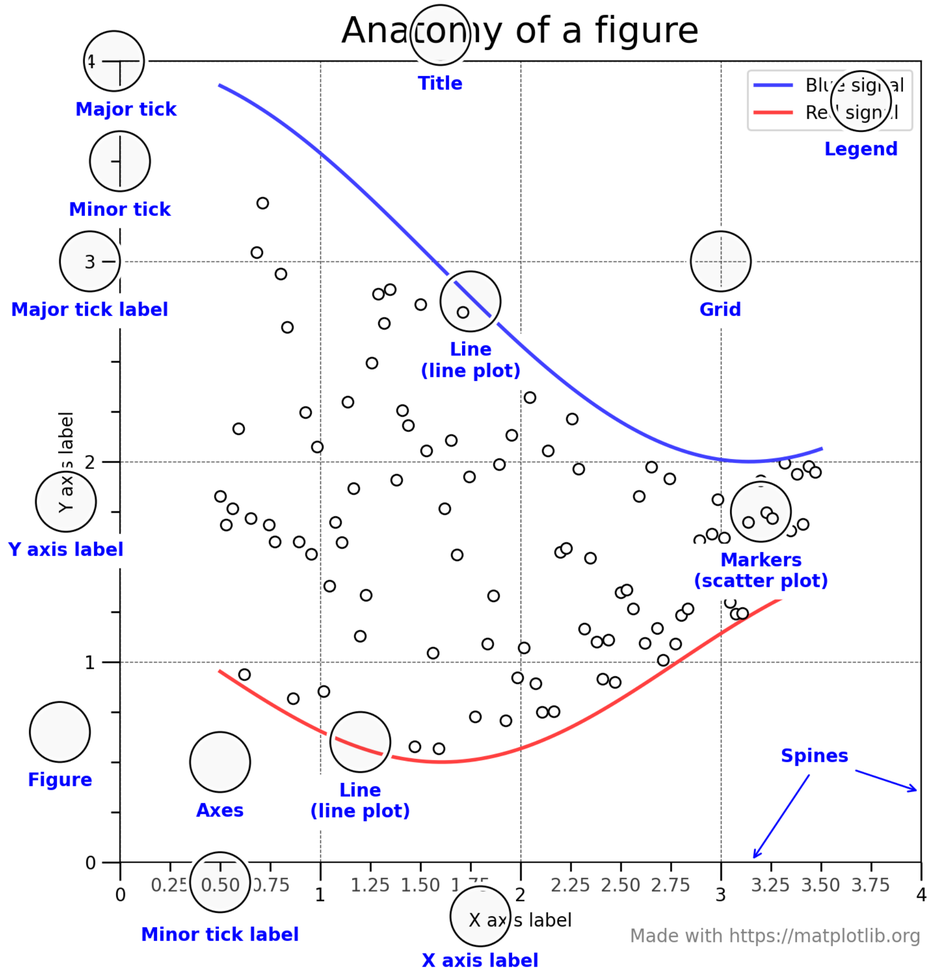


Matplotlib



- versatile and powerful library that allows you to design high quality figures
- it allows you to modify anything within a figure
- the timeline on the previous slide was created with matplotlib

Anatomy of a Figure



- a matplotlib figure is composed of a hierarchy of elements that form the actual figure
- Figure: the figure itself
  - you can specify size, background color and title
- Axes: the actual area where your data will be rendered; surrounded by spines (left, right, top, bottom)
- Axis: the decorated spines are called axis; x-axis (horizontal) and y-axis (vertical)
- Spines: the lines connecting the axis tick marks noting the boundaries of the data area; may be visible or invisible
- Artist: everything on the figure; when the figure is rendered, all of the artists are drawn to the canvas

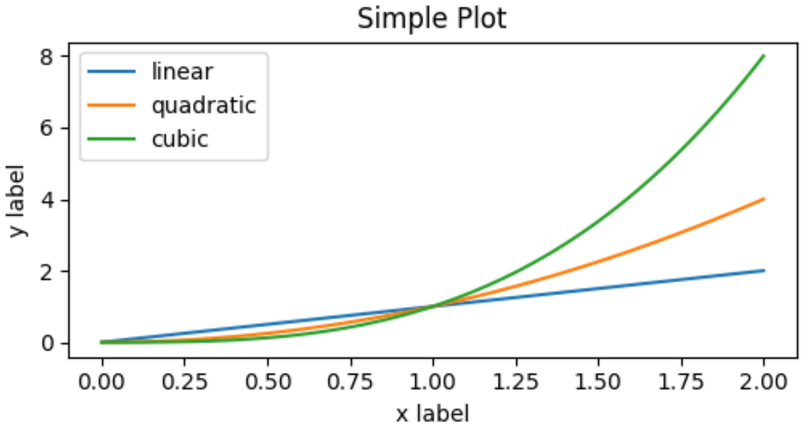
Coding Styles

```
x = np.linspace(0, 2, 100)  # Sample data.


fig, ax = plt.subplots(figsize=(5, 2.7), layout='constrained')
ax.plot(x, x, label='linear')  # Plot some data on the axes.
ax.plot(x, x**2, label='quadratic')  # Plot more data on the axes...
ax.plot(x, x**3, label='cubic')  # ... and some more.
ax.set_xlabel('x label')  # Add an x-label to the axes.
ax.set_ylabel('y label')  # Add a y-label to the axes.
ax.set_title("Simple Plot")  # Add a title to the axes.
ax.legend();  # Add a legend.
```

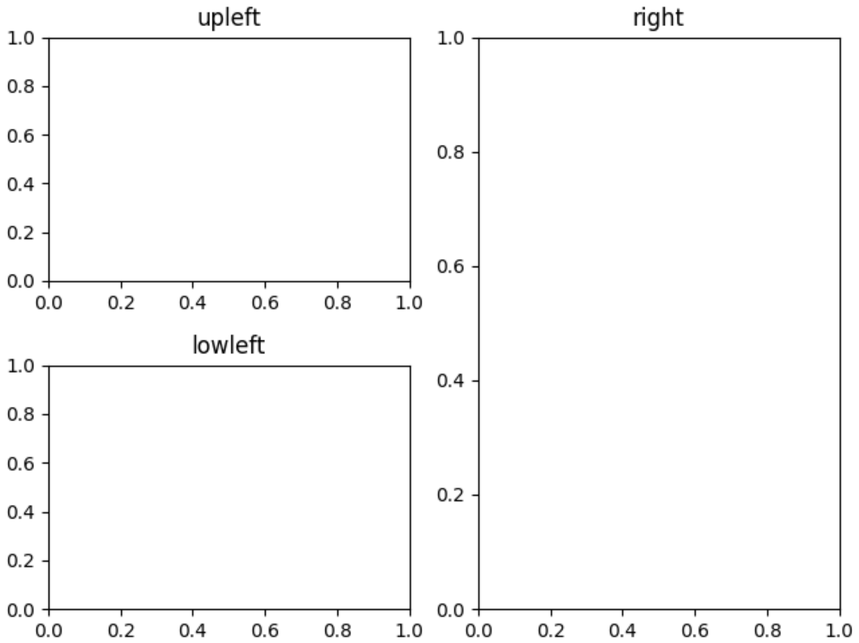
Object Oriented

```
x = np.linspace(0, 2, 100)  # Sample data.

plt.figure(figsize=(5, 2.7), layout='constrained')
plt.plot(x, x, label='linear')  # Plot some data on the (implicit) axes.
plt.plot(x, x**2, label='quadratic')  # etc.
plt.plot(x, x**3, label='cubic')
plt.xlabel('x label')
plt.ylabel('y label')
plt.title("Simple Plot")
plt.legend();
```

Pyplot style



Multiple Axes



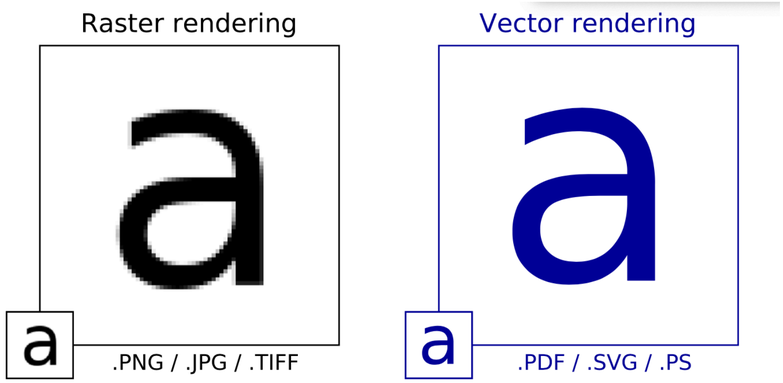
- a figure can have more than one Axes on it
- if you want your axes to be on a regular grid system, then use `plt.subplots(…)` to create a figure and add the axes to it automatically

```
fig, axes = plt.subplots(nrows=2, ncols=2)
```

```
fig, axd = plt.subplot_mosaic([['upleft', 'right'],
                               ['lowleft', 'right']], layout='constrained')
axd['upleft'].set_title('upleft')
axd['lowleft'].set_title('lowleft')
axd['right'].set_title('right');
```

Backends

Renderer	Type	Filetype
Agg	raster	Portable Network Graphic (PNG)
PS	vector	Postscript (PS)
PDF	vector	Portable Document Format (PDF)
SVG	vector	Scalable Vector Graphics (SVG)
Cairo	raster / vector	PNG / PDF / SVG




```
import matplotlib
matplotlib.use("renderer")
```

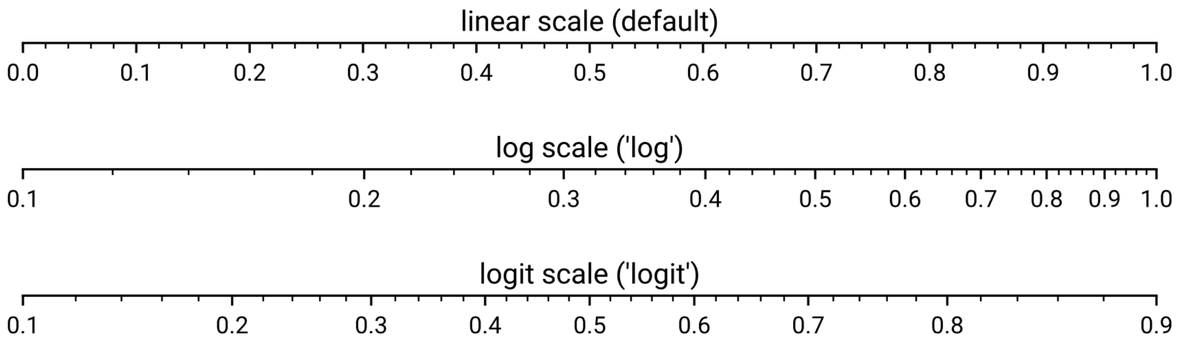
Saving Figures

```
x = np.linspace(0, 2, 100)


fig, ax = plt.subplots(figsize=(5, 2.7))
ax.plot(x, x, label='linear')
ax.plot(x, x**2, label='quadratic')
ax.plot(x, x**3, label='cubic')
ax.set_xlabel('x label')
ax.set_ylabel('y label')
ax.set_title("Simple Plot")
ax.legend()
fig.savefig("my_figure.png")
```

Available file formats (depending on installed backends):

- eps
- jpeg, jpg
- pdf
- png
- ps
- svg
- tif, tiff
- …

Scales



Define your own scale

```
def forward(x):
  return x**(1/2)

def inverse(x):
  return x**2

ax.set_xscale('function', functions=(forward, inverse))
```

- scales provide a mapping mechanism between the data and their representation
- matplotlib offers 4 different scales:
  - linear, log, symlog and logit
- each scale can be applied to x-axis only, y-axis only or to both
- log scales are used for values that are strictly positive
- the logit scale is used for values in (0,1) and uses a logarithmic scale on the border and quasi-linear scale in the middle (around 0.5)

Typography

- typography for visualizations means typeface, scripts, unicode, hinting, shaping, kerning, weight, slant …
- it is important to have a basic understanding of typography
    - read at least *Typography in ten minutes* by M. Butterick
- the matplotlib fontstack is defined using four different typeface families: sans, serif, monospace and cursive
- the default font stack is based on the DejaVu fonts

Fonts

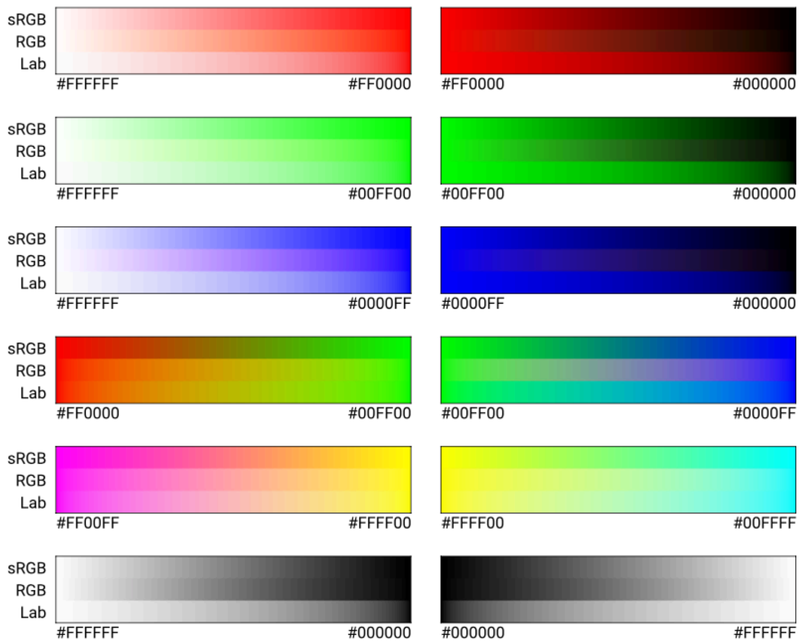
```
from matplotlib.font_manager import findfont, FontProperties
for family in ["serif", "sans", "monospace", "cursive"]:
  font = findfont(FontProperties(family=family))
  print(family, ":" , os.path.basename(font))
```

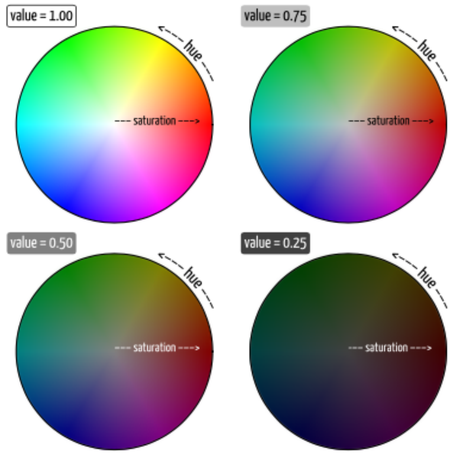
```
serif ; DejaVuSerif.ttf
sans ; DejaVuSans.ttf
monospace ; DejaVuSansMono.ttf
cursive ; Apple Chancery.ttf
```

Serif DejaVuSerif.ttf	Sans DejaVuSans.ttf	Monospace DejaVuSansMono.ttf	*Cursive* Apple Chancery.ttf
Serif RobotoSlab-Regular.ttf	Sans RobotoCondensed-Regular.ttf	Monospace RobotoMono-Regular.ttf	*Cursive* Merienda-Regular.ttf
Serif SourceSerifPro-Regular.otf	Sans SourceSansPro-Regular.ttf	Monospace SourceCodePro-Regular.ttf	*Cursive* ITC Zapf Chancery.ttf

- you can design your own font stack by choosing a set of alternative font families
- the default can be changed by modifying the rcParams or the stylesheets

Colors





HSV Color wheel

- to represent color on a computer we use a color model (how do we represent color) and a color space (what colors can be represented)
- color models: RGB, HSV, CMYK, …
- color spaces: sRGB, Adobe RGB, …
  - sRGB is the standard color space; mix different amounts of red, green and blue light to obtain a color of your choice
- when you specify a color in matplotlib, it is encoded in the sRGB color model and space

Open Colors

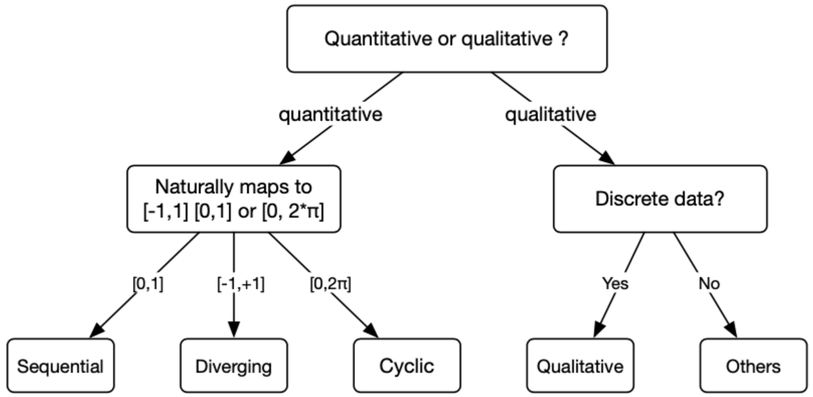
ORANGE 0 #FFF4E6	ORANGE 1 #FFE8CC	ORANGE 2 #FFD8A8	ORANGE 3 #FFC078	ORANGE 4 #FFA94D	ORANGE 5 #FF922B	ORANGE 6 #FD7E14	ORANGE 7 #F76707	ORANGE 8 #E8590C	ORANGE 9 #D9480F
YELLOW 0 #FFF9DB	YELLOW 1 #FFF3BF	YELLOW 2 #FFEC99	YELLOW 3 #FFE066	YELLOW 4 #FFD43B	YELLOW 5 #FCC419	YELLOW 6 #FAB005	YELLOW 7 #F59F00	YELLOW 8 #F08C00	YELLOW 9 #E67700
LIME 0 #F4FCE3	LIME 1 #E9FAC8	LIME 2 #D8F5A2	LIME 3 #C0EB75	LIME 4 #A9E34B	LIME 5 #94D82D	LIME 6 #82C91E	LIME 7 #74B816	LIME 8 #66A80F	LIME 9 #5C940D
GREEN 0 #EBFBEE	GREEN 1 #D3F9D8	GREEN 2 #B2F2BB	GREEN 3 #8CE99A	GREEN 4 #69DB7C	GREEN 5 #51CF66	GREEN 6 #40C057	GREEN 7 #37B24D	GREEN 8 #2F9E44	GREEN 9 #2B8A3E
TEAL 0 #E6FCF5	TEAL 1 #C3FAE8	TEAL 2 #96F2D7	TEAL 3 #63E6BE	TEAL 4 #38D9A9	TEAL 5 #20C997	TEAL 6 #12B886	TEAL 7 #0CA678	TEAL 8 #099268	TEAL 9 #087F5B
CYAN 0 #E3FAFC	CYAN 1 #C5F6FA	CYAN 2 #99E9F2	CYAN 3 #66D9E8	CYAN 4 #3BC9DB	CYAN 5 #22B8CF	CYAN 6 #15AABF	CYAN 7 #1098AD	CYAN 8 #0C8599	CYAN 9 #0B7285
BLUE 0 #E7F5FF	BLUE 1 #D0EBFF	BLUE 2 #A5D8FF	BLUE 3 #74C0FC	BLUE 4 #4DABF7	BLUE 5 #339AF0	BLUE 6 #228BE6	BLUE 7 #1C7ED6	BLUE 8 #1971C2	BLUE 9 #1864AB
INDIGO 0 #EDF2FF	INDIGO 1 #DBE4FF	INDIGO 2 #BAC8FF	INDIGO 3 #91A7FF	INDIGO 4 #748FFC	INDIGO 5 #5C7CFA	INDIGO 6 #4C6EF5	INDIGO 7 #4263EB	INDIGO 8 #3B5BDB	INDIGO 9 #364FC7
VIOLET 0 #F3F0FF	VIOLET 1 #E5DBFF	VIOLET 2 #D0BFFF	VIOLET 3 #B197FC	VIOLET 4 #9775FA	VIOLET 5 #845EF7	VIOLET 6 #7950F2	VIOLET 7 #7048E8	VIOLET 8 #6741D9	VIOLET 9 #5F3DC4
GRAPE 0 #F8F0FC	GRAPE 1 #F3D9FA	GRAPE 2 #EEBEFA	GRAPE 3 #E599F7	GRAPE 4 #DA77F2	GRAPE 5 #CC5DE8	GRAPE 6 #BE4BDB	GRAPE 7 #AE3EC9	GRAPE 8 #9C36B5	GRAPE 9 #862E9C
PINK 0 #FFF0F6	PINK 1 #FFDEEB	PINK 2 #FCC2D7	PINK 3 #FAA2C1	PINK 4 #F783AC	PINK 5 #F06595	PINK 6 #E64980	PINK 7 #D6336C	PINK 8 #C2255C	PINK 9 #A61E4D
RED 0 #FFF5F5	RED 1 #FFE3E3	RED 2 #FFC9C9	RED 3 #FFA8A8	RED 4 #FF8787	RED 5 #FF6B6B	RED 6 #FA5252	RED 7 #F03E3E	RED 8 #E03131	RED 9 #C92A2A
GRAY 0 #F8F9FA	GRAY 1 #F1F3F5	GRAY 2 #E9ECEF	GRAY 3 #DEE2E6	GRAY 4 #CED4DA	GRAY 5 #ADB5BD	GRAY 6 #868E96	GRAY 7 #495057	GRAY 8 #343A40	GRAY 9 #212529

Material Colors

BLUE GREY 0 #ECEFF1	BLUE GREY 1 #CFD8DC	BLUE GREY 2 #B0BEC5	BLUE GREY 3 #90A4AE	BLUE GREY 4 #78909C	BLUE GREY 5 #607D8B	BLUE GREY 6 #546E7A	BLUE GREY 7 #455A64	BLUE GREY 8 #37474F	BLUE GREY 9 #263238
GREY 0 #FAFAFA	GREY 1 #F5F5F5	GREY 2 #EEEEEE	GREY 3 #E0E0E0	GREY 4 #BDBDBD	GREY 5 #9E9E9E	GREY 6 #757575	GREY 7 #616161	GREY 8 #424242	GREY 9 #212121
BROWN 0 #EFEBE9	BROWN 1 #D7CCC8	BROWN 2 #BCAAA4	BROWN 3 #A1887F	BROWN 4 #8D6E63	BROWN 5 #795548	BROWN 6 #6D4C41	BROWN 7 #5D4037	BROWN 8 #4E342E	BROWN 9 #3E2723
D.ORANGE 0 #FBE9E7	D.ORANGE 1 #FFCCBC	D.ORANGE 2 #FFAB91	D.ORANGE 3 #FF8A65	D.ORANGE 4 #FF7043	D.ORANGE 5 #FF5722	D.ORANGE 6 #F4511E	D.ORANGE 7 #E64A19	D.ORANGE 8 #D84315	D.ORANGE 9 #BF360C
ORANGE 0 #FFF3E0	ORANGE 1 #FFE0B2	ORANGE 2 #FFCC80	ORANGE 3 #FFB74D	ORANGE 4 #FFA726	ORANGE 5 #FF9800	ORANGE 6 #FB8C00	ORANGE 7 #F57C00	ORANGE 8 #EF6C00	ORANGE 9 #E65100
AMBER 0 #FFF8E1	AMBER 1 #FFECB3	AMBER 2 #FFE082	AMBER 3 #FFD54F	AMBER 4 #FFCA28	AMBER 5 #FFC107	AMBER 6 #FFB300	AMBER 7 #FFA000	AMBER 8 #FF8F00	AMBER 9 #FF6F00
YELLOW 0 #FFFDE7	YELLOW 1 #FFF9C4	YELLOW 2 #FFF59D	YELLOW 3 #FFF176	YELLOW 4 #FFEE58	YELLOW 5 #FFEB3B	YELLOW 6 #FDD835	YELLOW 7 #FBC02D	YELLOW 8 #F9A825	YELLOW 9 #F57F17
LIME 0 #F9FBE7	LIME 1 #F0F4C3	LIME 2 #E6EE9C	LIME 3 #DCE775	LIME 4 #D4E157	LIME 5 #CDDC39	LIME 6 #C0CA33	LIME 7 #AFB42B	LIME 8 #9E9D24	LIME 9 #827717
L.GREEN 0 #F1F8E9	L.GREEN 1 #DCEDC8	L.GREEN 2 #C5E1A5	L.GREEN 3 #AED581	L.GREEN 4 #9CCC65	L.GREEN 5 #8BC34A	L.GREEN 6 #7CB342	L.GREEN 7 #689F38	L.GREEN 8 #558B2F	L.GREEN 9 #33691E
GREEN 0 #E8F5E9	GREEN 1 #C8E6C9	GREEN 2 #A5D6A7	GREEN 3 #81C784	GREEN 4 #66BB6A	GREEN 5 #4CAF50	GREEN 6 #43A047	GREEN 7 #388E3C	GREEN 8 #2E7D32	GREEN 9 #1B5E20
TEAL 0 #E0F2F1	TEAL 1 #B2DFDB	TEAL 2 #80CBC4	TEAL 3 #4DB6AC	TEAL 4 #26A69A	TEAL 5 #009688	TEAL 6 #00897B	TEAL 7 #00796B	TEAL 8 #00695C	TEAL 9 #004D40
CYAN 0 #E0F7FA	CYAN 1 #B2EBF2	CYAN 2 #80DEEA	CYAN 3 #4DD0E1	CYAN 4 #26C6DA	CYAN 5 #00BCD4	CYAN 6 #00ACC1	CYAN 7 #0097A7	CYAN 8 #00838F	CYAN 9 #006064
L.BLUE 0 #E1F5FE	L.BLUE 1 #B3E5FC	L.BLUE 2 #81D4FA	L.BLUE 3 #4FC3F7	L.BLUE 4 #29B6F6	L.BLUE 5 #03A9F4	L.BLUE 6 #039BE5	L.BLUE 7 #0288D1	L.BLUE 8 #0277BD	L.BLUE 9 #01579B
BLUE 0 #E3F2FD	BLUE 1 #BBDEFB	BLUE 2 #90CAF9	BLUE 3 #64B5F6	BLUE 4 #42A5F5	BLUE 5 #2196F3	BLUE 6 #1E88E5	BLUE 7 #1976D2	BLUE 8 #1565C0	BLUE 9 #0D47A1
INDIGO 0 #E8EAF6	INDIGO 1 #C5CAE9	INDIGO 2 #9FA8DA	INDIGO 3 #7986CB	INDIGO 4 #5C6BC0	INDIGO 5 #3F51B5	INDIGO 6 #3949AB	INDIGO 7 #303F9F	INDIGO 8 #283593	INDIGO 9 #1A237E
D.PURPLE 0 #EDE7F6	D.PURPLE 1 #D1C4E9	D.PURPLE 2 #B39DDB	D.PURPLE 3 #9575CD	D.PURPLE 4 #7E57C2	D.PURPLE 5 #673AB7	D.PURPLE 6 #5E35B1	D.PURPLE 7 #512DA8	D.PURPLE 8 #4527A0	D.PURPLE 9 #311B92
PURPLE 0 #F3E5F5	PURPLE 1 #E1BEE7	PURPLE 2 #CE93D8	PURPLE 3 #BA68C8	PURPLE 4 #AB47BC	PURPLE 5 #9C27B0	PURPLE 6 #8E24AA	PURPLE 7 #7B1FA2	PURPLE 8 #6A1B9A	PURPLE 9 #4A148C
PINK 0 #FCE4EC	PINK 1 #F8BBD0	PINK 2 #F48FB1	PINK 3 #F06292	PINK 4 #EC407A	PINK 5 #E91E63	PINK 6 #D81B60	PINK 7 #C2185B	PINK 8 #AD1457	PINK 9 #880E4F
RED 0 #FFEBEE	RED 1 #FFCDD2	RED 2 #EF9A9A	RED 3 #E57373	RED 4 #EF5350	RED 5 #F44336	RED 6 #E53935	RED 7 #D32F2F	RED 8 #C62828	RED 9 #B71C1C

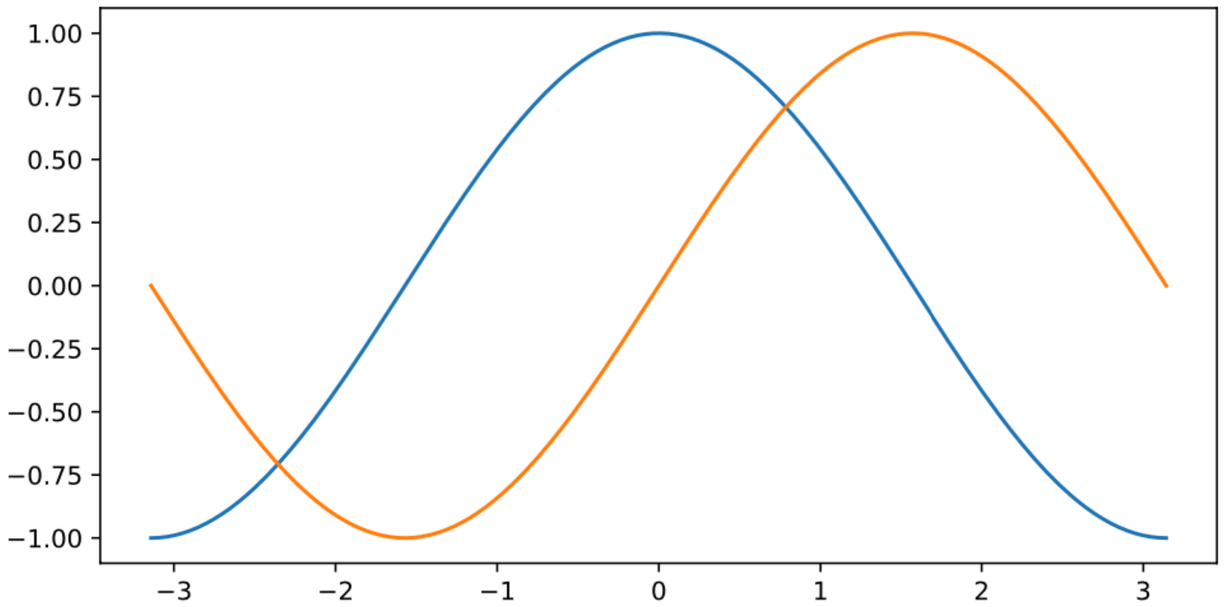
Color Maps

- colormapping corresponds to the mapping of values to colors
- it is important to use the right colormap that corresponds to your data
- How to choose the right colormap?

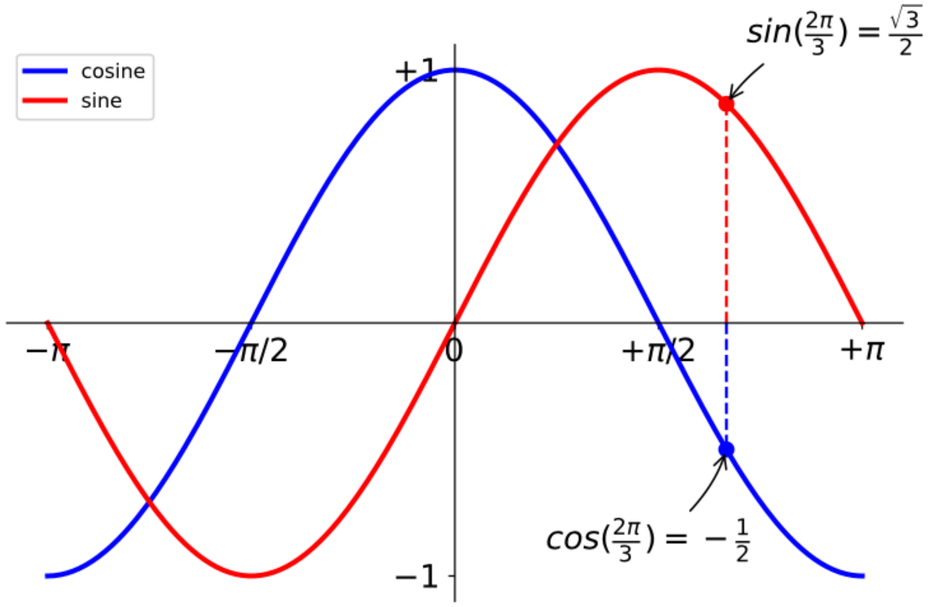


Don’t Trust the Defaults

DEFAULT

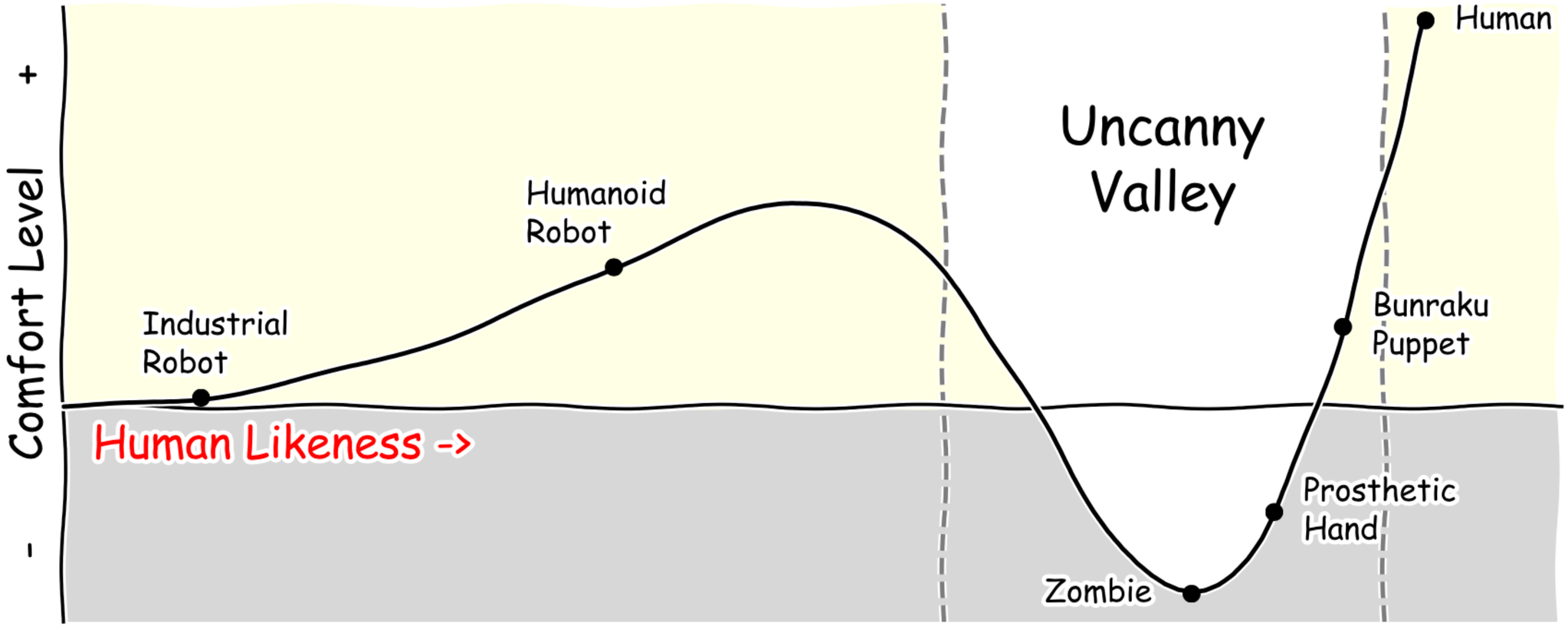


IMPROVED



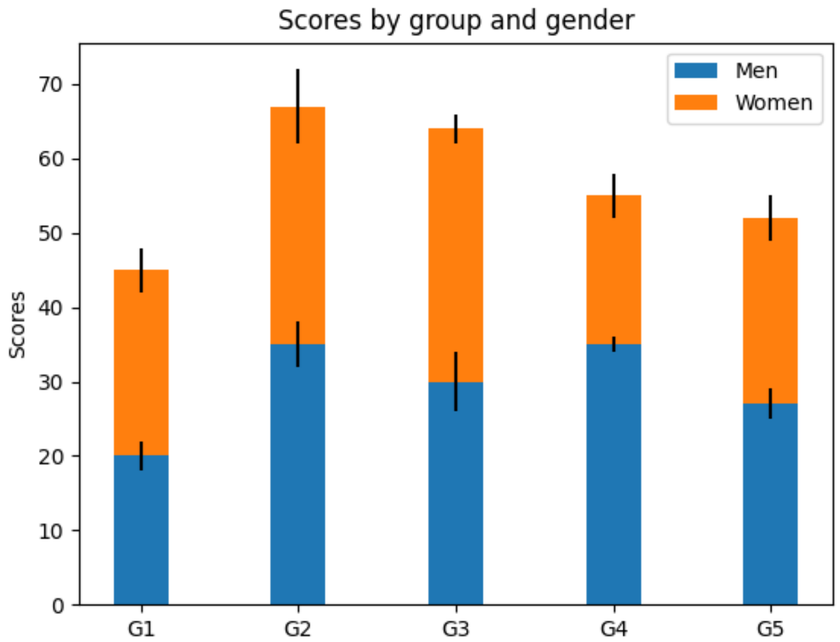
HOCHSCHULE LANDSHUT

Message Trumps Beauty



Examples

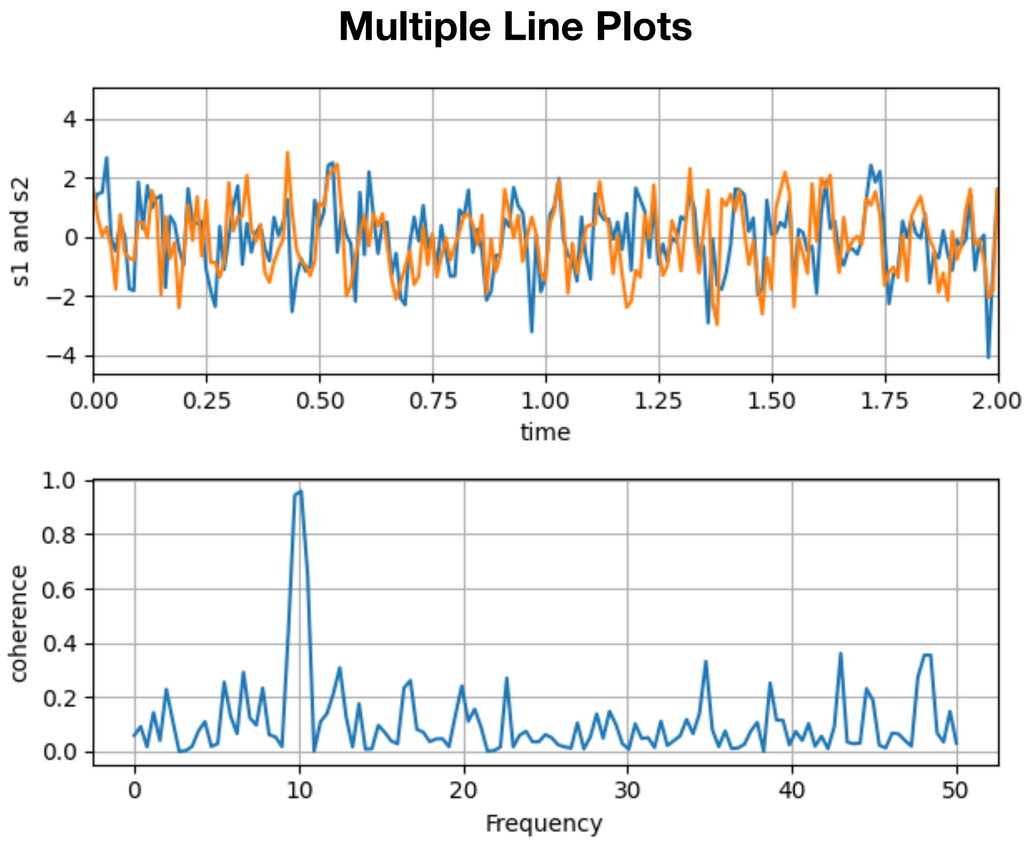
Stacked Bar Chart



Source: matplotlib gallery

```
import matplotlib.pyplot as plt


labels = ['G1', 'G2', 'G3', 'G4', 'G5']
men_means = [20, 35, 30, 35, 27]
women_means = [25, 32, 34, 20, 25]
men_std = [2, 3, 4, 1, 2]
women_std = [3, 5, 2, 3, 3]
width = 0.35 # the width of the bars: can also be len(x) sequence

fig, ax = plt.subplots()

ax.bar(labels, men_means, width, yerr=men_std, label='Men')
ax.bar(labels, women_means, width, yerr=women_std, bottom=men_means,
       label='Women')

ax.set_ylabel('Scores')
ax.set_title('Scores by group and gender')
ax.legend()

plt.show()
```

Examples

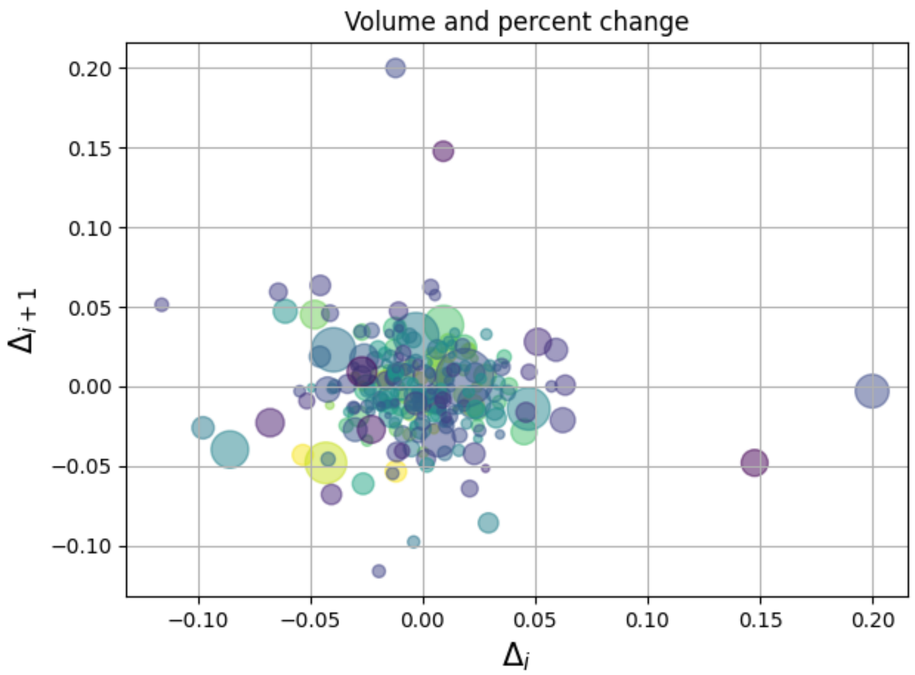


Source: matplotlib gallery

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

# Fixing random state for reproducibility
np.random.seed(19680801)

dt = 0.01
t = np.arange(0, 30, dt)
nse1 = np.random.randn(len(t))         # white noise 1
nse2 = np.random.randn(len(t))         # white noise 2

# Two signals with a coherent part at 10Hz and a random part
s1 = np.sin(2 * np.pi * 10 * t) + nse1
s2 = np.sin(2 * np.pi * 10 * t) + nse2

fig, axs = plt.subplots(2, 1)
axs[0].plot(t, s1, t, s2)
axs[0].set_xlim(0, 2)
axs[0].set_xlabel('time')
axs[0].set_ylabel('s1 and s2')
axs[0].grid(True)

cxy, f = axs[1].cohere(s1, s2, 256, 1. / dt)
axs[1].set_ylabel('coherence')

fig.tight_layout()
plt.show()
```

Examples

Scatter Plot



Source: matplotlib gallery

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


price_data = (cbook.get_sample_data('goog.npz', np_load=True)['price_data']
              .view(np.recarray))
price_data = price_data[-250:]  # get the most recent 250 trading days

delta1 = np.diff(price_data.adj_close) / price_data.adj_close[:-1]

# Marker size in units of points^2
volume = (15 * price_data.volume[:-2] / price_data.volume[0])**2
close = 0.003 * price_data.close[:-2] / 0.003 * price_data.open[:-2]

fig, ax = plt.subplots()
ax.scatter(delta1[:-1], delta1[1:], c=close, s=volume, alpha=0.5)

ax.set_xlabel(r'$\Delta_i$', fontsize=15)
ax.set_ylabel(r'$\Delta_{i+1}$', fontsize=15)
ax.set_title('Volume and percent change')

ax.grid(True)
fig.tight_layout()

plt.show()
```

Your Turn

- read and work through the following matplotlib tutorials:
  - basic usage
  - pyplot
  - the lifecycle of a plot
- try to recreate the right plot on slide 16 (sin and cosine functions) without the formula annotations

seaborn

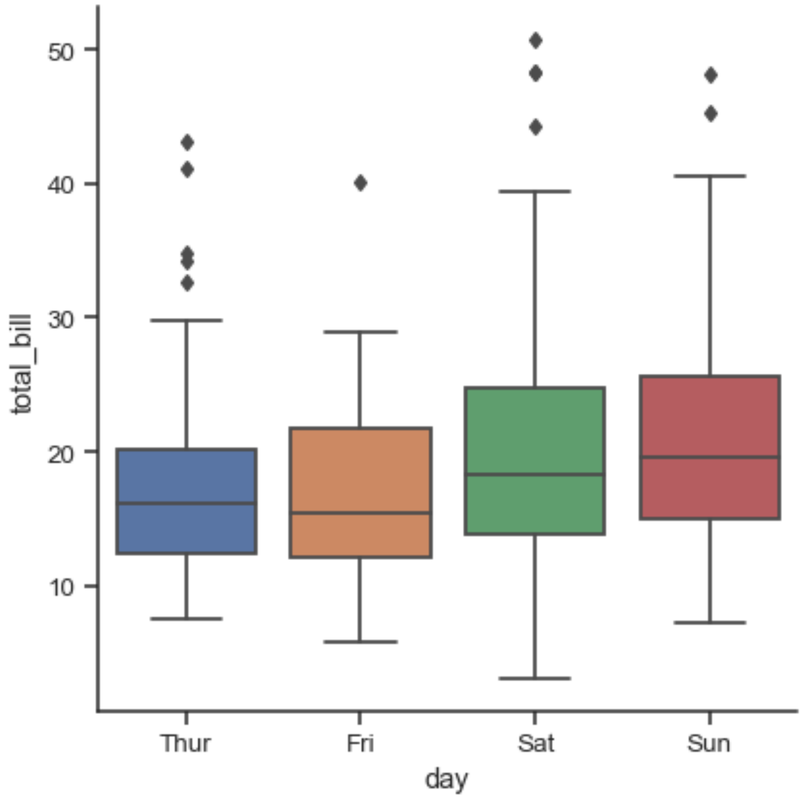
seaborn

- if you are a little overwhelmed by the customizability of matplotlib, the seaborn library might be a good choice to start your data visualization journey

- it is based on matplotlib and provides a high-level interface for drawing attractive and informative statistical graphics

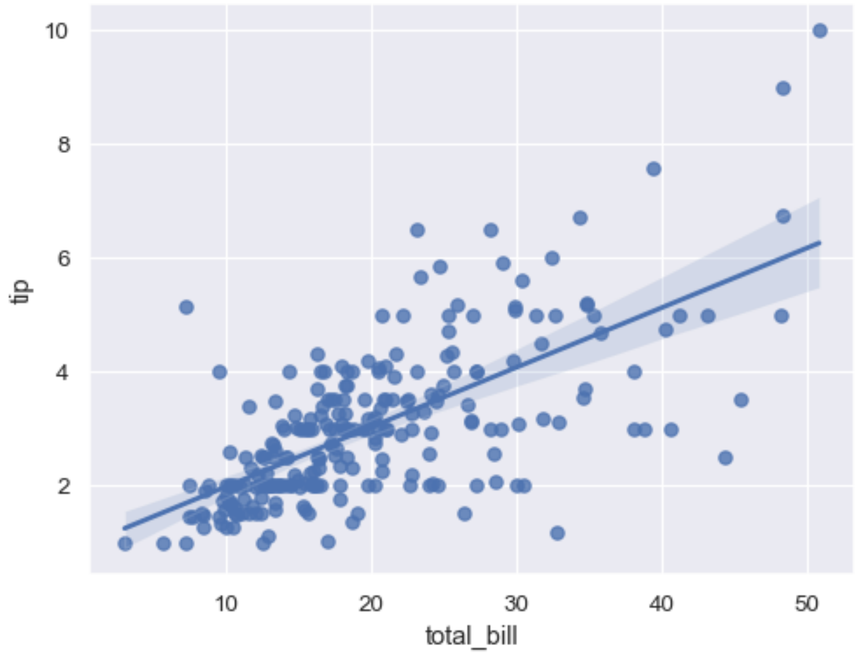
- installation: `conda install seaborn`

seaborn examples



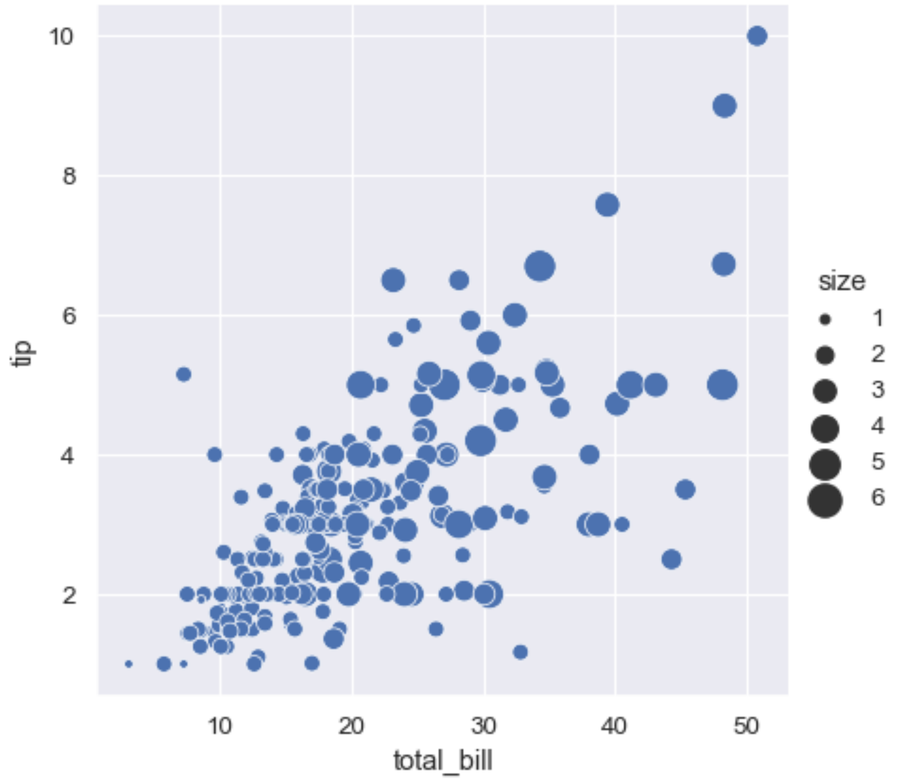

```
import seaborn as sns
sns.set_theme(style="ticks", color_codes=True)

tips = sns.load_dataset("tips")
sns.catplot(x="day", y="total_bill", kind="box", data=tips)
```

seaborn examples




```
import seaborn as sns
sns.set_theme(style="ticks", color_codes=True)

tips = sns.load_dataset("tips")
sns.regplot(x="total_bill", y="tip", data=tips);
```

seaborn examples




```
import seaborn as sns
sns.set_theme(style="ticks", color_codes=True)

tips = sns.load_dataset("tips")
sns.relplot(x="total_bill", y="tip", size="size", sizes=(15, 200), data=tips);
```

References

- Slide 2,5: N.P. Rougier, Scientific Visualization - Python & Matplotlib
- Slide 6: Wikipedia: https://en.wikipedia.org/wiki/Black_hole
- Slide 7: Matplotlib Tutorial: https://matplotlib.org/stable/tutorials/introductory/usage.html#sphx-glr-tutorials-introductory-usage-py
- Slide 9,10,12,14-19: N.P. Rougier, Scientific Visualization - Python & Matplotlib
- Slide 20: Matplotlib Gallery: https://matplotlib.org/devdocs/gallery/lines_bars_and_markers/bar_stacked.html#sphx-glr-gallery-lines-bars-and-markers-bar-stacked-py
- Slide 21: Matplotlib Gallery: https://matplotlib.org/devdocs/gallery/lines_bars_and_markers/cohere.html#sphx-glr-gallery-lines-bars-and-markers-cohere-py
- Slide 22: Matplotlib Gallery: https://matplotlib.org/devdocs/gallery/lines_bars_and_markers/scatter_demo2.html#sphx-glr-gallery-lines-bars-and-markers-scatter-demo2-py