Next-Level Data Visualization in Python

A practical guide to upgrading your plots
by making the most of matplotlib (and more)

ODSC East 2023

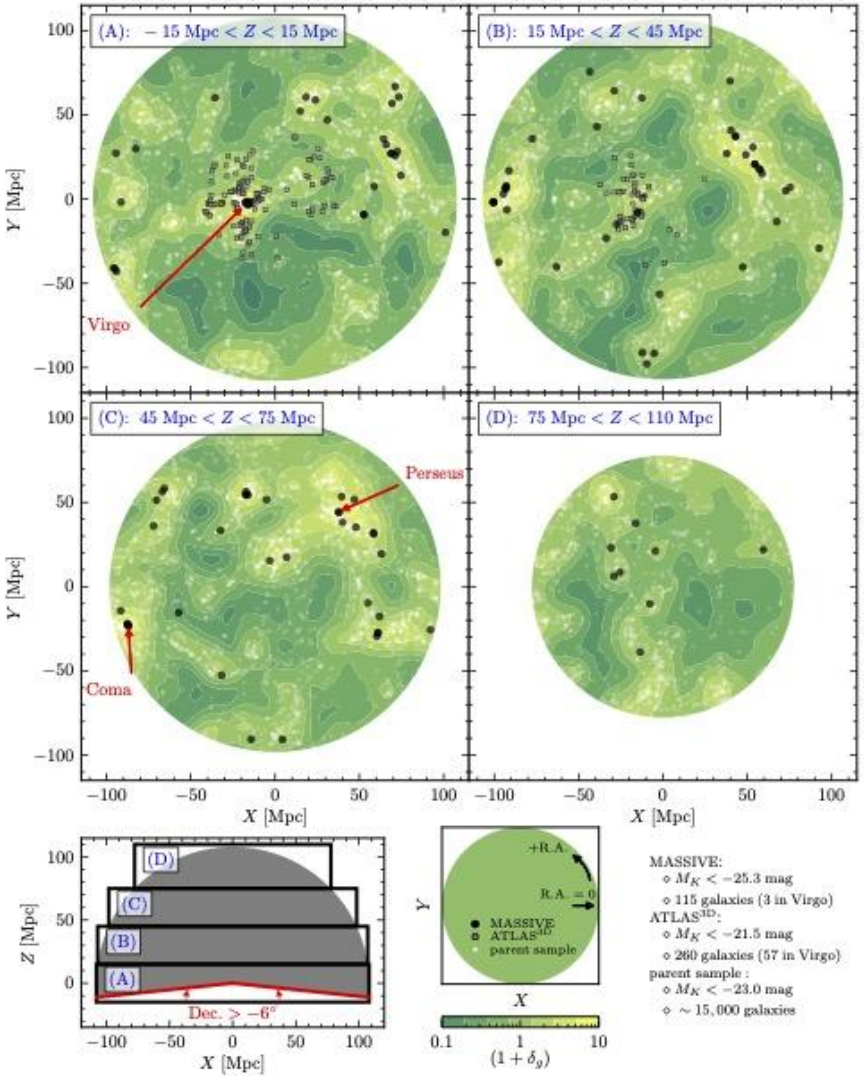
Melanie Veale

Who am I?

- 2011-2018
  Astrophysics Ph.D. and postdoc
  at University of California, Berkeley
  (data analytics with python)

- 2019-2023
  Field Data Scientist
  at Domino Data Lab
  (cloud ML platform, customer success)

- Feb 2023-present
  Data Solutions Architect
  at Anomalo
  (data quality platform, customer success)



Tutorial Format

Plotting sample data: US Census educational attainment by age and sex
https://github.com/melanieveale/odsc-east-2023-python-plots-plus
Running notebooks yourself as we go is optional, but encouraged!

(slides) → (notebooks demo) → (Q&A) → (repeat x3)

Agenda

Sections:

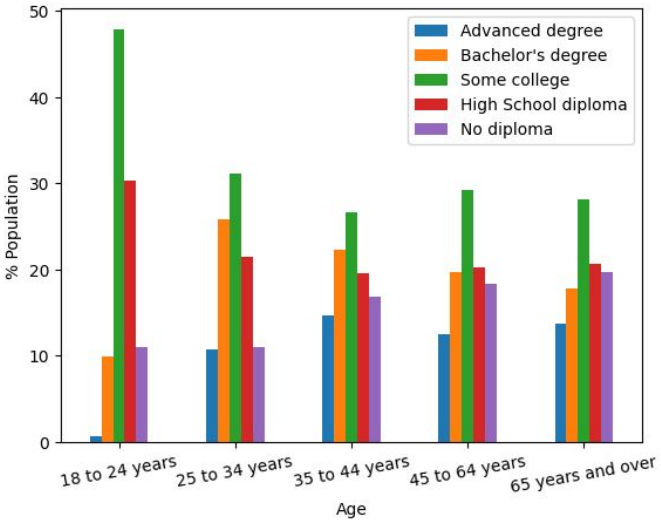
1. Introduction and fundamentals
2. Education level by age
3. Population by county
4. (Bonus) Creating maps

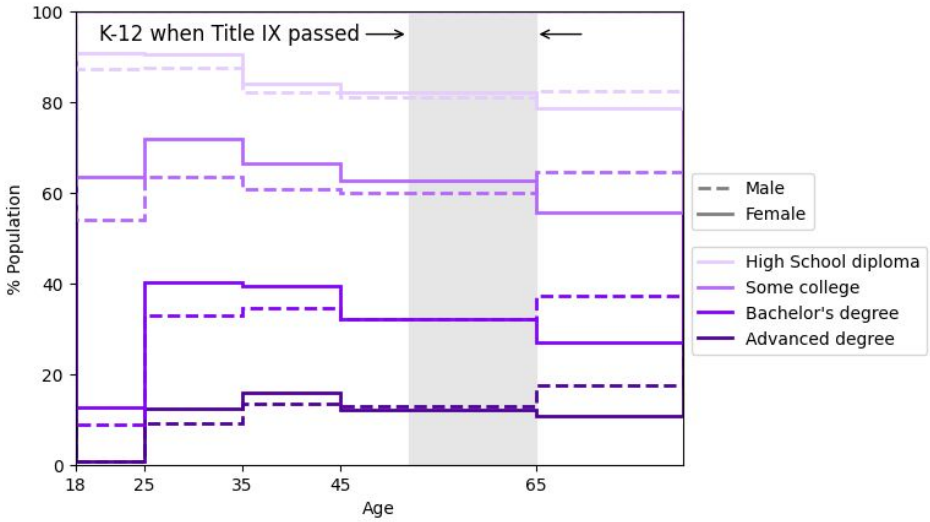
Topics:

- Axes limits, labels, and scales
- Use (and misuse) of color
- Annotation and labeling
- Layouts and other formatting
- Reusable code

Goals

20% good data storytelling practices / 80% how to implement them





Goals

“I’m lost in the wilderness of documentation and stackoverflow, send help!”

✅ Wilderness survival tips

✅ Go deep on a few examples

✅ Understanding plot code structure

✅ Working with third-party libraries

✅ Publication quality images

🚫 Showing you a single clear path

🚫 Covering every common example

🚫 Data analysis best practices

🚫 Non-matplotlib-based libraries

🚫 Interactive visualizations

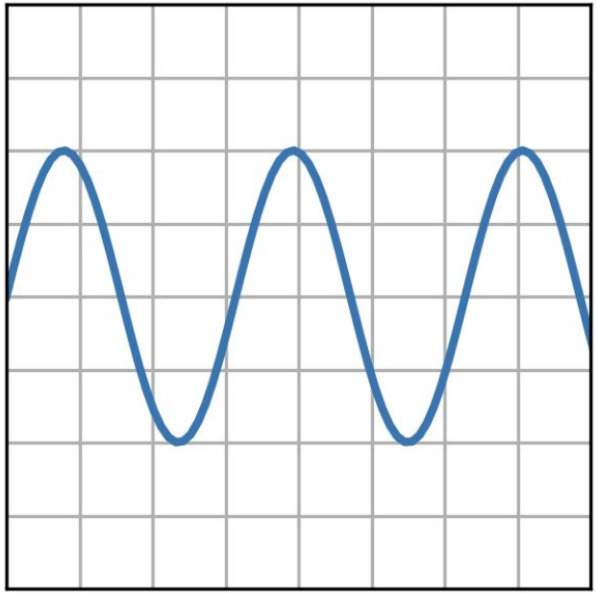
Section 1: Fundamentals

Matplotlib history and basics

https://matplotlib.org/

“Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. **Matplotlib makes easy things easy and hard things possible.**”

“A large number of third party packages extend and build on Matplotlib functionality, including several higher-level plotting interfaces (seaborn, HoloViews, ggplot, …), and a projection and mapping toolkit (Cartopy).”



plot(x, y)

Matplotlib history and basics

https://en.wikipedia.org/wiki/Matplotlib

Open-source, Python, with MATLAB inspiration

2003 (John Hunter) → 2013 (v1.3.0, shared copyright) → today (v3.7.1)

Used by Pandas for plotting

Go-to for many for static plots, but not the only option

Has some stiff competition for interactive plots (not covered today)

Matplotlib fundamentals: implicit vs explicit

Implicit

```
plt.bar(df['x'], df['x'])
plt.xlabel('X label')
plt.ylabel('Y label')
```

Explicit

```
fig, ax = plt.subplots()
ax.bar(df['x'], df['y'])
ax.set_xlabel('X label')
ax.set_ylabel('Y label')
```

- Implicit (**pyplot**) interface is fewer lines of code for quick and dirty plots.
- Explicit (**object oriented**) interface gives finer control (recommended)

What about pandas df.plot()?

DataFrames in Pandas have a built-in plotting method to handle most plots.

This returns a standard Matplotlib object, allowing mixing and matching, which gives you the best of both worlds!

```
df.plot(‘x’, ‘y’,
  kind=’bar’,
  xlabel=’X label’,
  ylabel=’Y label’)
```

⇨

```
ax = df.plot('x', 'y',
  kind=’bar’)
ax.set_xlabel('X label')
ax.set_ylabel('Y label')
```

Exporting plots and the “tight” layout

Follow the documentation for details about file format, resolution, etc.

To solve the problem of margins being cut off when saving:

```
fig.savefig('my-fig.png', bbox_inches = 'tight')
plt.savefig(‘my-fig.png’, bbox_inches = 'tight')
```

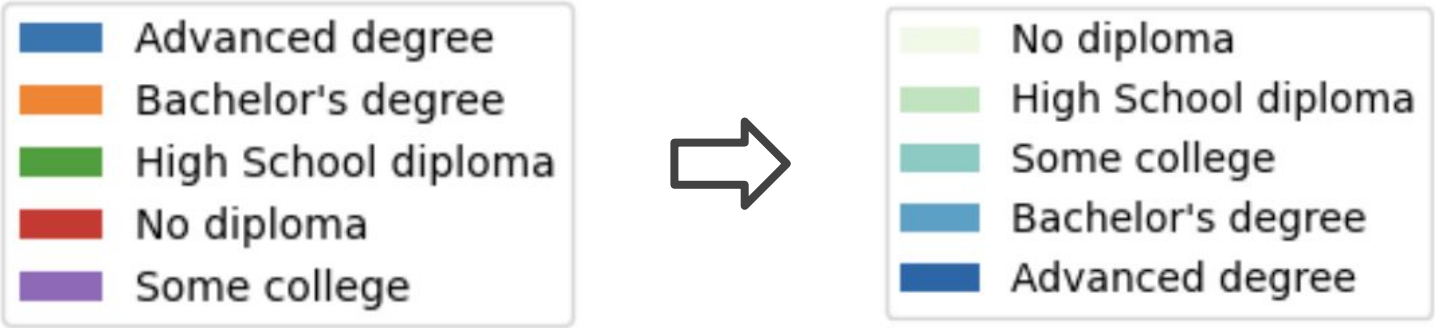
(More advanced layout control in the bonus section, time permitting!)

Notebook 1

Q&A

Section 2:
Education level by age

Color part 1



- Basic color theory is very useful; always consider colorblindness!
- Refer to documentation for ways to specify color (names, RGB, hex, etc)
- Experiment with online tools like this, or this and it's variations
- Here we use a **sequential** color scheme for (sorted) **ordinal** categories

Wilderness survival tip: extra kwargs

Example:
Pandas documentation says nothing about the `color` kwarg...

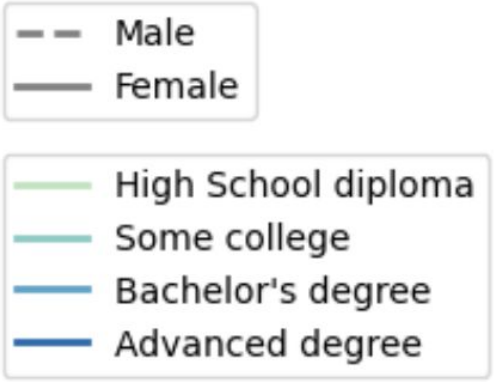
```
df2.plot(
    'x', 'y', kind='bar',
    color=mycolors)
```

```
plt.bar(
    df['x'], df['y'],
    color=mycolors)
```

...because it just passes it to the matplotlib function under the hood.

So, even when using pandas, you may want to reference the matplotlib docs!
It's also a good reason to plot with matplotlib directly in development mode.

All about legends



This will be our first major exercise that requires full OO matplotlib code.

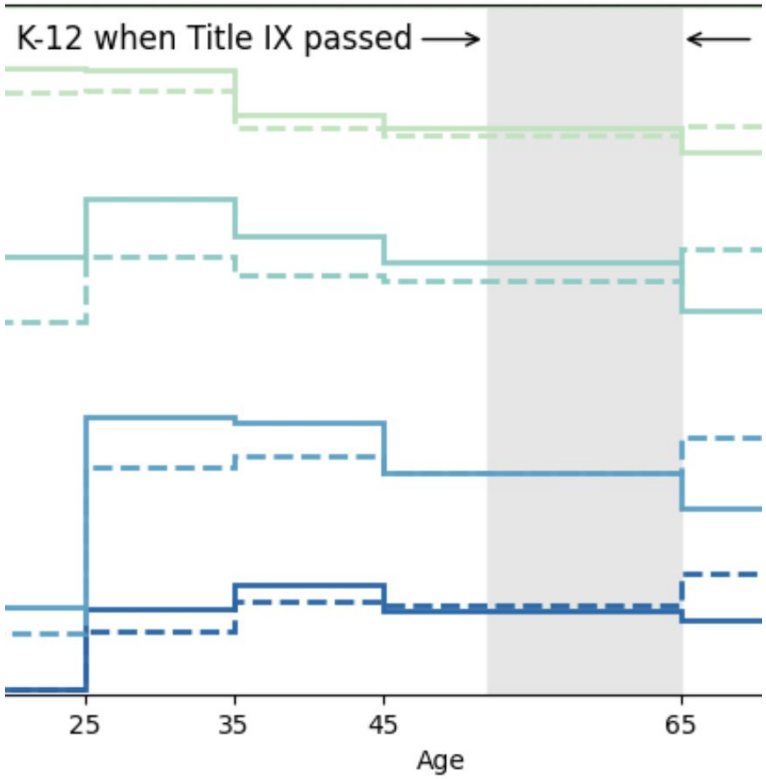
For reference: read the legend guide!**

**Do it after you have tasted success with this example, to avoid scaring yourself too much with the complexity!

Annotation

Annotations are comparatively straightforward in the docs...
but you MUST think like a designer to avoid cluttered plots!

```
ax.annotate(
  mytext, xy=(52, 95), xytext=(47,95),
  ha='right', fontsize='large',
  arrowprops=dict(facecolor='k',
    arrowstyle='->'))
ax.axvspan(52, 65, color='0.9', zorder=-1)
```



Notebook 2

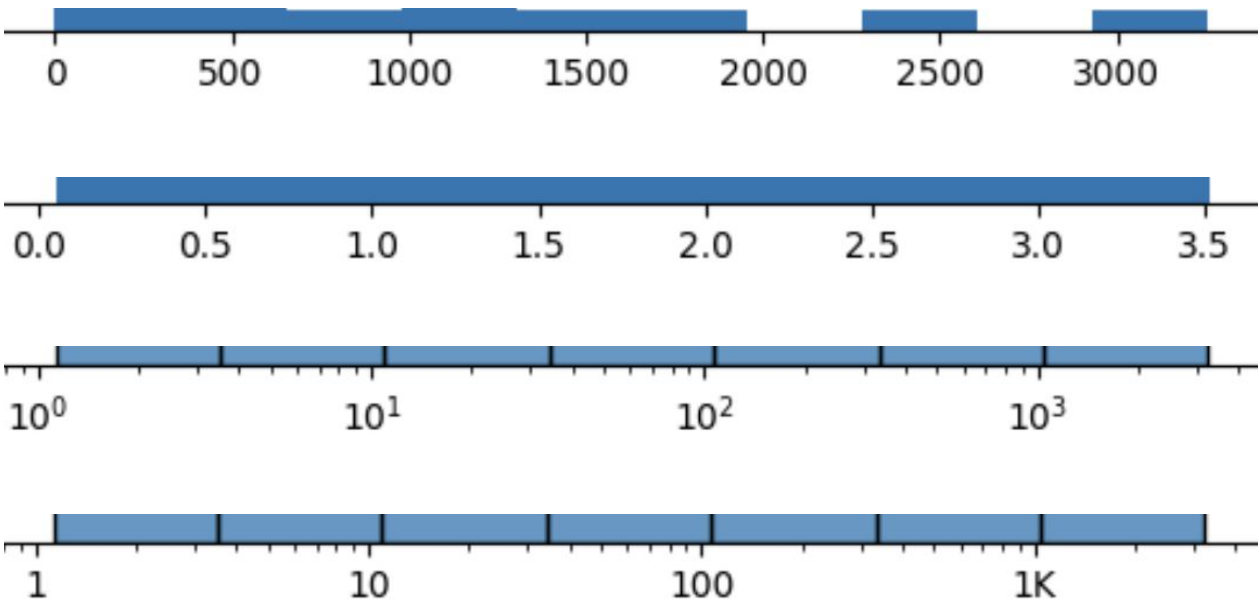
Q&A

Section 3:
Population by county

Log scales

All axes on the right represent the same underlying data.

Our quest for the ideal log scale axis will touch on all the topics in this section.



Third-party libraries

Pandas is not the only third-party library that makes extensive use of matplotlib for plotting.

Seaborn happens to have a good option for log-scale histograms, as well as attractive default styles.

Apply what you learn here to many of the third-party libraries!

The current list: some of these are tiny extensions, and some are very full-featured libraries themselves.

HoloViews,hvPlot,**pandas**,plotnine,Xarray,animatplot,celluloid,gif,manimplotlib,numpngw,xmovie,Aquarel,CMasher,cmcrameri,cmocean,cmyt,Colorcet,distinctipy,Farrow&Ball Matplotlib,mplcyberpunk,TUEplots,viscm,Sphinx-Gallery,ArviZ,**Astropy**,BGHeatmaps,colorio,cplot,DNAFeaturesViewer,grplot,HockeyRink,librosa,MetPy,microfilm,mir_eval,mplfinance,mplhep,mplsignal,mplsoccer,MyForestPlot,NetworkX,planetMagFields,Py-ART,pyCircos,pyGenomeViz,pymatviz,PyPlutchik,**seaborn**,seaborn-image,Yellowbrick,mpl-gui,mpl-qtthread,Glue,Lumen,mpl-multitab,Pylustrator,PyNanoLab,PySimpleGUI,sview-gui,mpl-draggable-line,mpl-image-label,mpl-image-segmenter,mpl-interactions,mpl-point-clicker,mpl_widget_box,mplcursors,mpldatacursor,mplinorm,Panel,Quibbler,cartopy,EOmaps,**GeoPandas**,geoplot,GeoViews,mplsteronet,prettymaps,ridge_map,matplotcheck,matplotx,Blume,Datashader,matplotlib-venn,mpl-scatter-density,plottable,PyUpSet,Windrose,yt,adjustText,brokenaxes,FigPager,Flexitext,grid-strategy,HighlightText,matplotlib-label-lines,matplotlib-scalebar,matplotview,mpl-probscale,mpl-template,MPLTable,patchworklib,ProPlot,skunk,svgpath2mpl,tikzplotlib,gr,mplcairo,wxmplot

Reusable code

```
ax.set_xticks([1, 10, 100, 10**3])
ax.set_xticklabels(["1", "10", "100", "1K"])
```

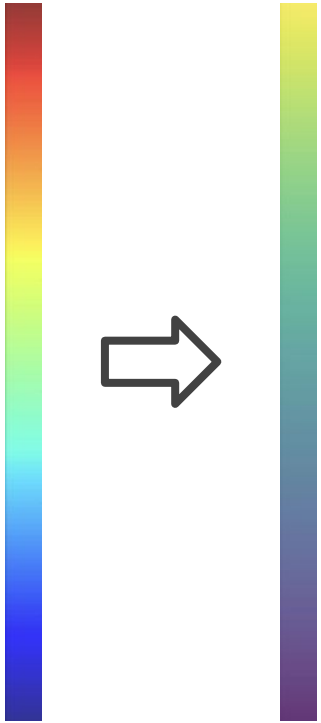
Looks simple, easy to copy!

Looks mysterious, prone to weird errors when trying to copy...

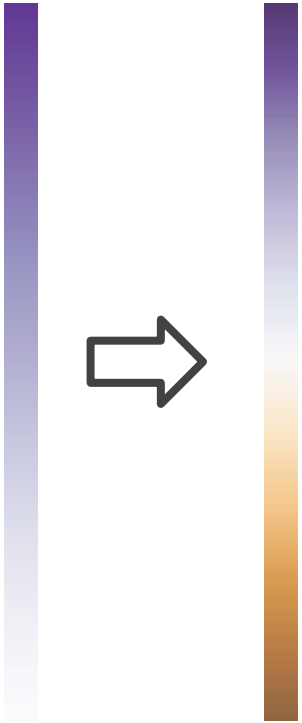
```
def myfunc(t, p)
    …lots more code…
ax.xaxis.set_major_formatter(myfunc)
```

So why ever use examples like the second one?
More work up front = less work to reuse your code later!

Color part 2



- Colormaps are a large topic, worth their own whole tutorial
- Many good ones are available today (see named options in the docs)
- Many bad ones have existed historically (see one of many rants against “jet”)
- Choose carefully based on the nature of your data



Wilderness survival tip: object classes

```
fig, ax = plt.subplots()
sc = ax.scatter(x, y, c=z)
cb = fig.colorbar(sc, ax=ax)
```

A very simple example for customizing a colorbar

Four separate object classes, each with many methods, tips, tricks, and possibilities

```
matplotlib.figure.Figure
matplotlib.axes._axes.Axes
matplotlib.collections.PathCollection
matplotlib.colorbar.Colorbar
```

Variable naming conventions give good hints - but beware mixing them up!

Notebook 3

Q&A

Section 4: Maps

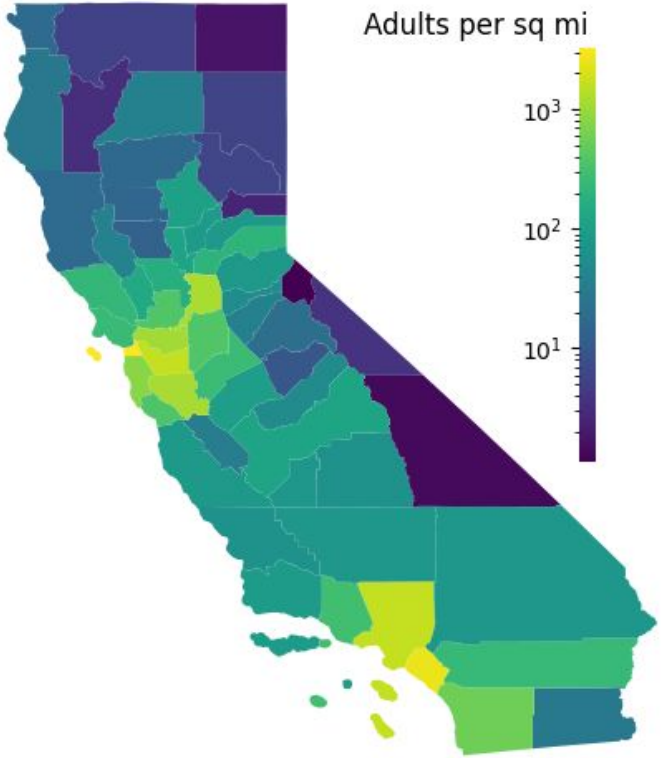
Maps and population data

We have one final bonus example:

Using **GeoPandas** to plot our population data on a map of California counties.

This is an example of a **chloropleth** map. There are important disadvantages to chloropleths, and mapping population data is particularly tricky, so beware.

But for any mapping library built on matplotlib - you are now more prepared!



Notebook 4

Closing comments

Summary

I hope you feel more prepared to navigate the wilderness of Matplotlib documentation and Stackoverflow examples!

Wilderness survival skills we covered today:

- Combining third-party library convenience with matplotlib raw customization power
- Navigating documentation, with a few key landmarks (especially color and legends)
- Connecting Matplotlib object classes to common variable naming conventions

Where to go next

Some suggested topics to hone your new skills with:

- Error bars and confidence bands
- Using `matplotlib.rcParams`
- Axes tips for time series data
- Twinned axes and grids of axes
- Density plots and contours
- Math rendering and all things LaTeX
- Visualization principles for big data
- And many more!

Q&A

Thank you!