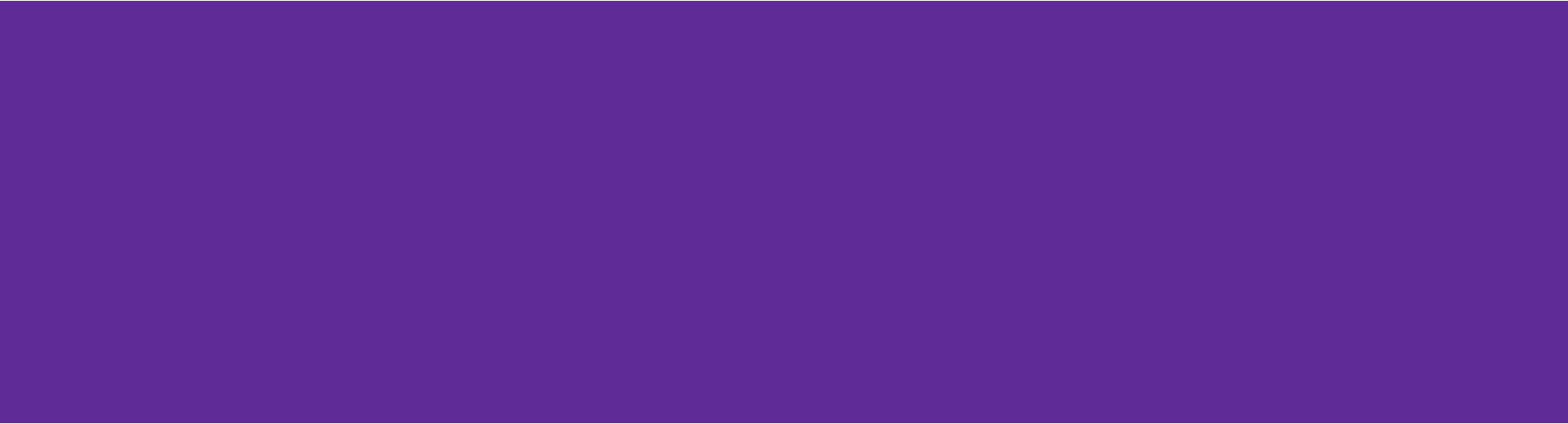


CS 1: Intro to CS

More Matplotlib: Implementing Different Plots, Student Examples in Research!



Agenda

So far:

- Transition between CSV processing and Matplotlib
- Identifying data science questions
- Breaking down datasets for Matplotlib plot arguments (Part C practice)

Today:

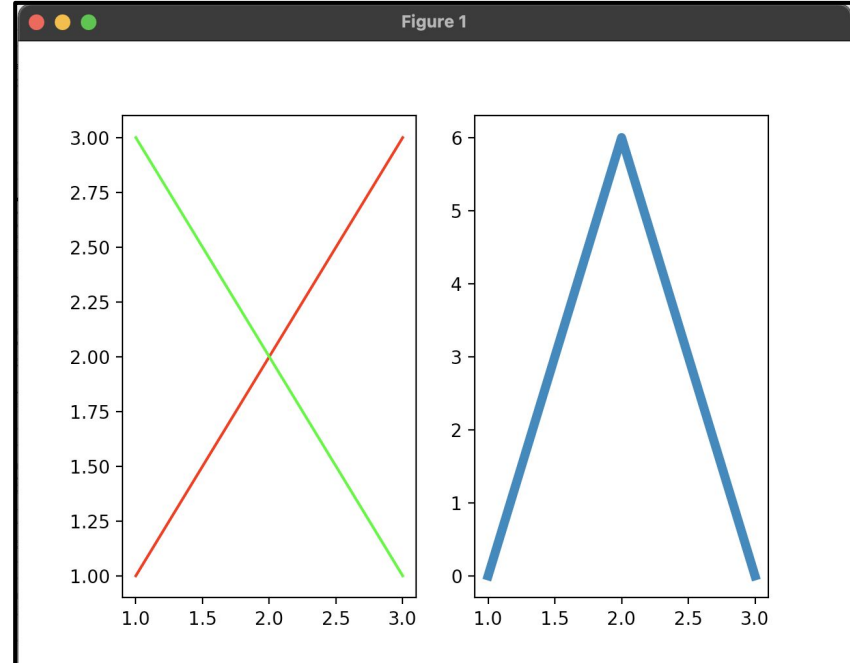
- Reviewing implicit vs. explicit approach and keyword arguments (previous slide deck)
- Implementing different plots: What's similar? What's different?
- Plotting options: titles/labels/axes/colors

(Optional materials on Matplotlib with NumPy included under Lecture 16)

Working with Multiple Plots: Basic Line Graphs

Once you have the **Axes** object(s), you can plot with them (`.plot` can be called on a single **Axes** multiple times). What do you think the arguments represent? What are default properties of a rendered plot?

```
# using the variable axs for multiple Axes
(fig, axs) = plt.subplots(1, 2)
(ax1, ax2) = axs
ax1.plot([1, 2, 3], [1, 2, 3],
         color='red')
ax1.plot([1, 2, 3], [3, 2, 1],
         color='#00ff00') # green
ax2.plot([1, 2, 3], [0, 3, 0],
         linewidth=5)
plt.show()
```



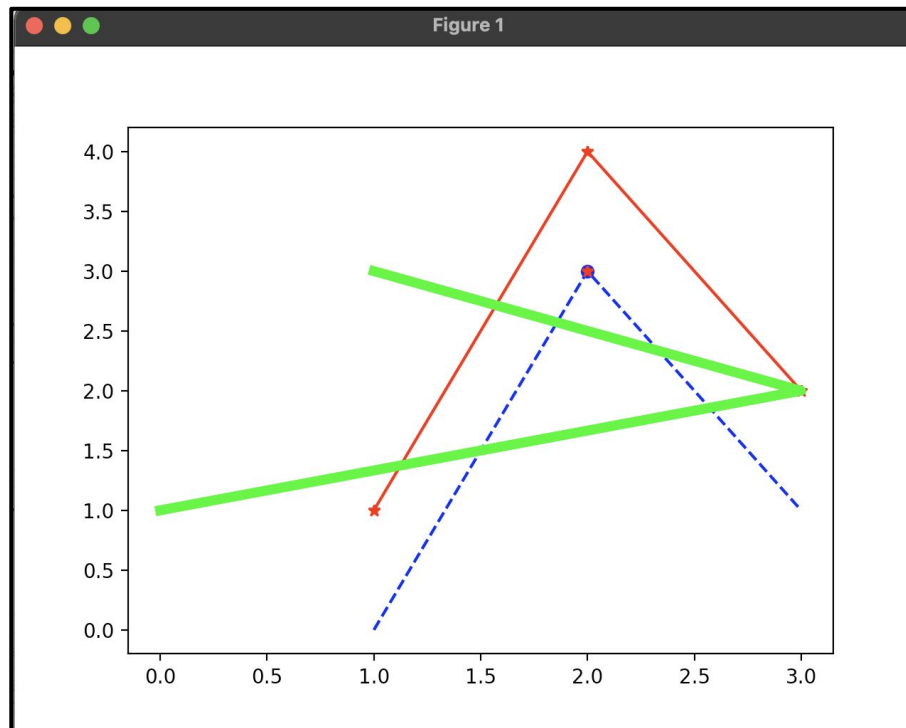
Plotting Lines (`ax.plot(x, y)`) vs. Points (`ax.plot(xs, ys)`)

```
xs = [1, 2, 3]
ys1 = [0, 3, 1]
ys2 = [1, 4, 2]
(fig, ax1) = plt.subplots()

# plot x and y using blue lines
ax1.plot(xs, ys1, 'b--')
# ditto, but with red stars
ax1.plot(xs, ys2, 'r*-')

(x, y) = (2, 3)
# plot (x, y) using blue circle markers
ax1.plot(x, y, 'bo')
# ditto, but with red stars
ax1.plot(x, y, 'r*')

ax1.plot(ys1, xs, color='#00ff00', linewidth=5)
ax1.plot(2, 3, color='#d14900') # Caltech orange!
```



Creating a Bar Chart: `ax.bar(xs, ys, [opts])`

```
songs = ['N95', '505', "World's Smallest Violin", 'Thnks fr th Mmrs',  
        'Headlines', 'West Coast', 'Shook Ones, Pt. II',  
        "Don't Wanna Fall In Love"]
```

```
popularities = [83, 82, 78, 78, 77, 76, 76, 73]
```

```
(fig, ax) = plt.subplots()
```

```
fig.set_size_inches(8, 7)
```

```
ax.bar(songs, popularities, color='blue', width=0.6)
```

```
# Abbreviate songs if we want to override tick labels
```

```
for i in range(len(songs)):
```

```
    abbr = songs[i]
```

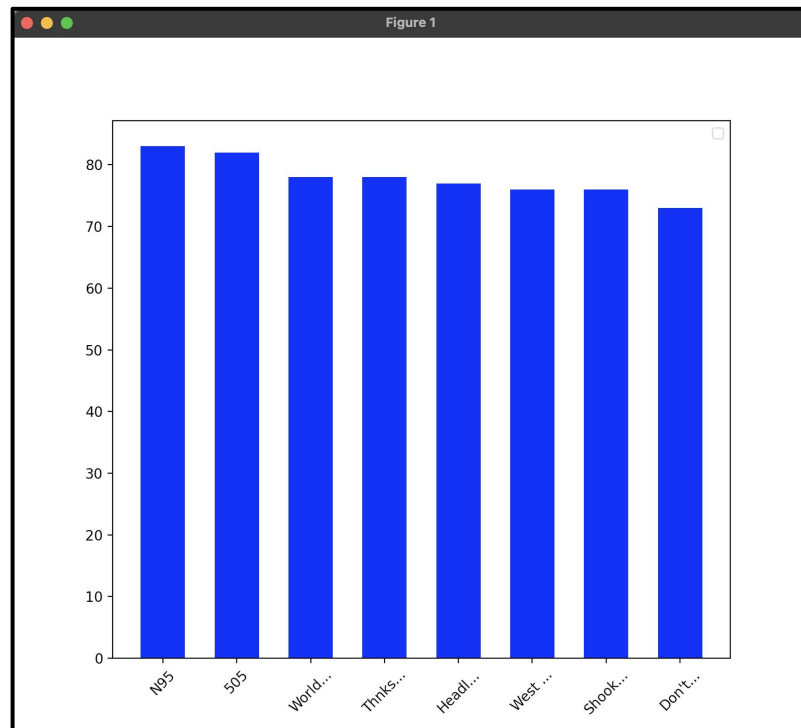
```
    if len(abbr) > 5:
```

```
        abbr = abbr[:5] + '...'
```

```
        songs[i] = abbr
```

```
ax.set_xticklabels(songs, rotation=45)
```

```
plt.show()
```



Labels and Plot Title

```
(fig, ax) = plt.subplot()
ax.bar(songs, popularities,
       color="blue", width=0.6)
ax.set_xlabel('Songs')
ax.set_ylabel('Popularity')
ax.set_title('Popularity by song')

# Remove x ticks/labels on x-axis
ax.xticks([])
```

```
plt.bar(songs, popularities,
        color="blue", width=0.6)
plt.xlabel('Songs')
plt.ylabel('Popularity')
plt.title('Popularity by song')

# Remove x ticks/labels on x-axis
plt.xticks([])
```

Note: The left is preferred to use so you can generalize your plotting code with multiple **Axes** (plots) in larger programs.

Choosing Plots

We won't go in depth about data visualization/statistics in this class, but here's a summary of how to choose an appropriate plot for your data

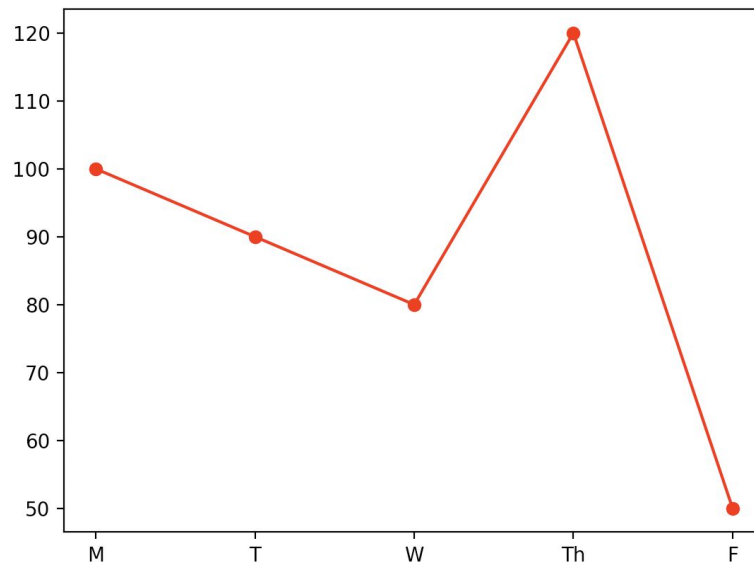
Line Plots

Used to represent a relationship between the x-axis and y-axis

In Matplotlib:

```
xs = ['M', 'T', 'W', 'Th', 'F']  
ys = [100, 90, 80, 120, 50]  
(fig, ax) = plt.subplots()  
ax.plot(xs, ys, 'ro-')  
plt.show()
```

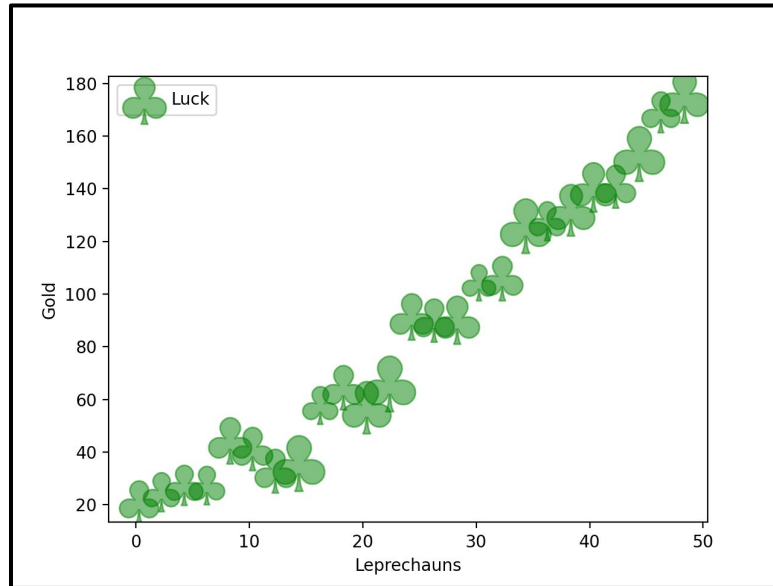
~



Scatter Plot

Used to show the relationship between variables represented as dots (no lines)

- The below example also shows how to set custom markers/styles



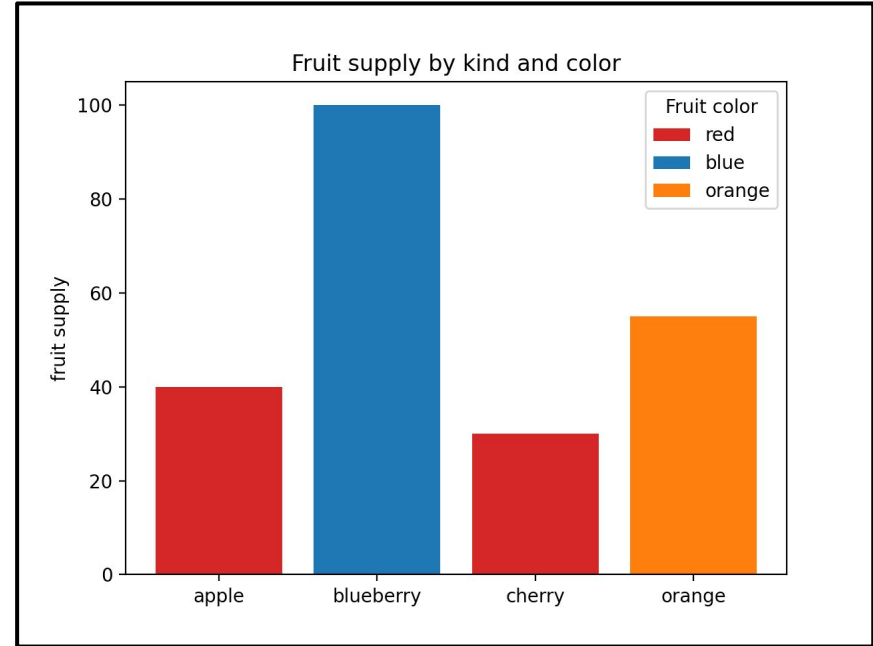
Source:

https://matplotlib.org/stable/gallery/lines_bars_and_markers/scatter_custom_symbol.html

Bar Chart

Used to show the relationship between numeric and categorical values

- A vertical bar chart has categorical values on x-axis
- A horizontal bar chart has categorical values on y-axis



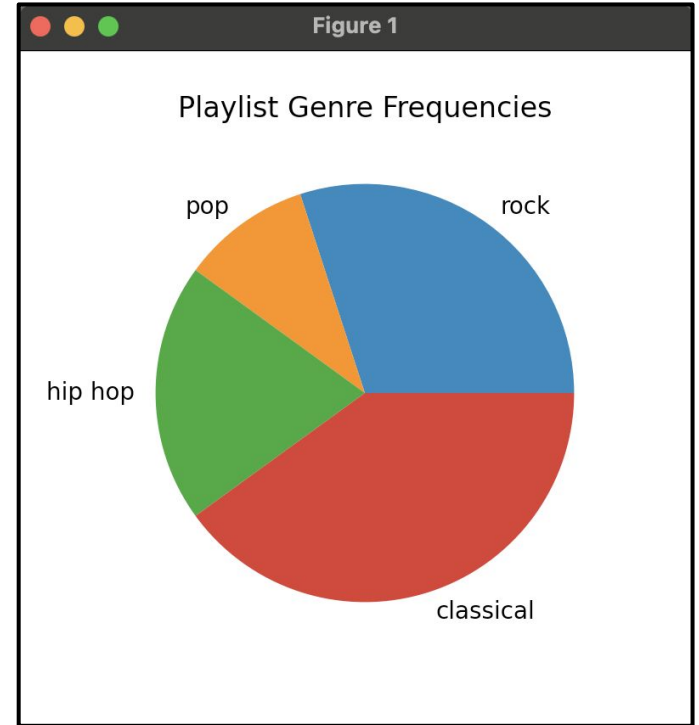
Source:

https://matplotlib.org/stable/gallery/lines_bars_and_markers/bar_colors.html

Pie Chart

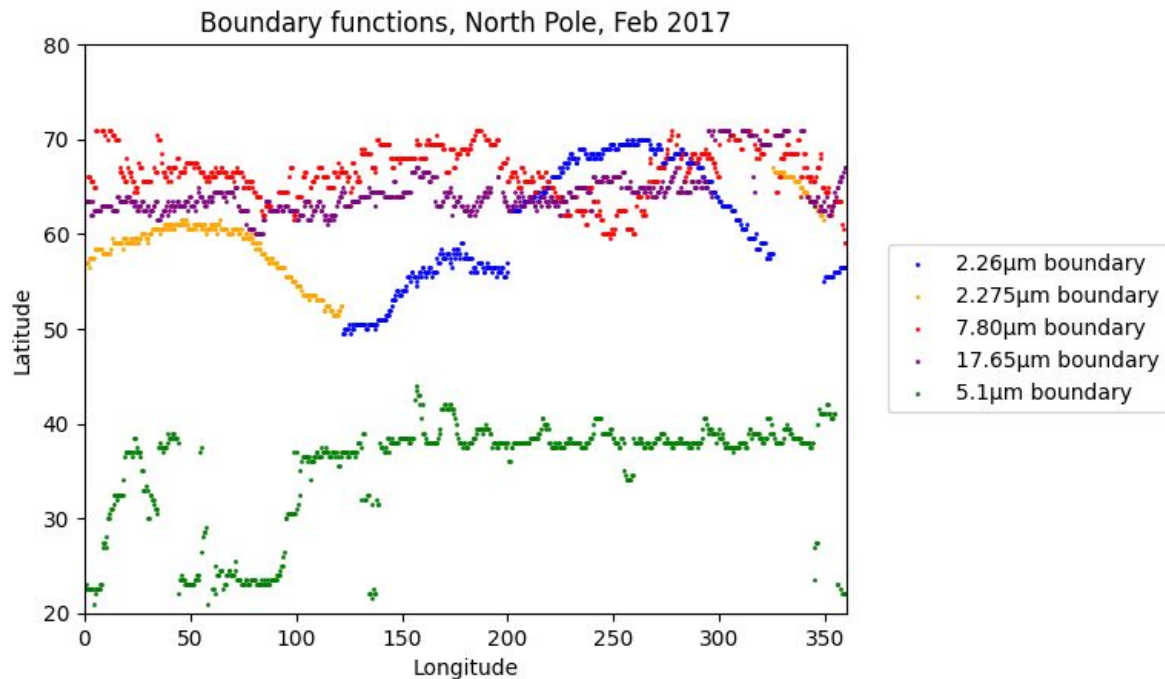
Used to show relative percentages of values

```
genre_frequencies = {'rock': 3, 'pop': 1,  
                    'hip hop': 2, 'classical': 4}  
labels = genre_frequencies.keys()  
frequencies = genre_frequencies.values()  
  
fig, ax1 = plt.subplots()  
fig.set_size_inches(4, 4)  
# In Lab04, we had a second Axes for the legend  
# for more control of space, but here we just  
# use labels  
ax1.pie(frequencies, labels=labels)  
ax1.set_title('Playlist Genre Frequencies')  
plt.show()
```



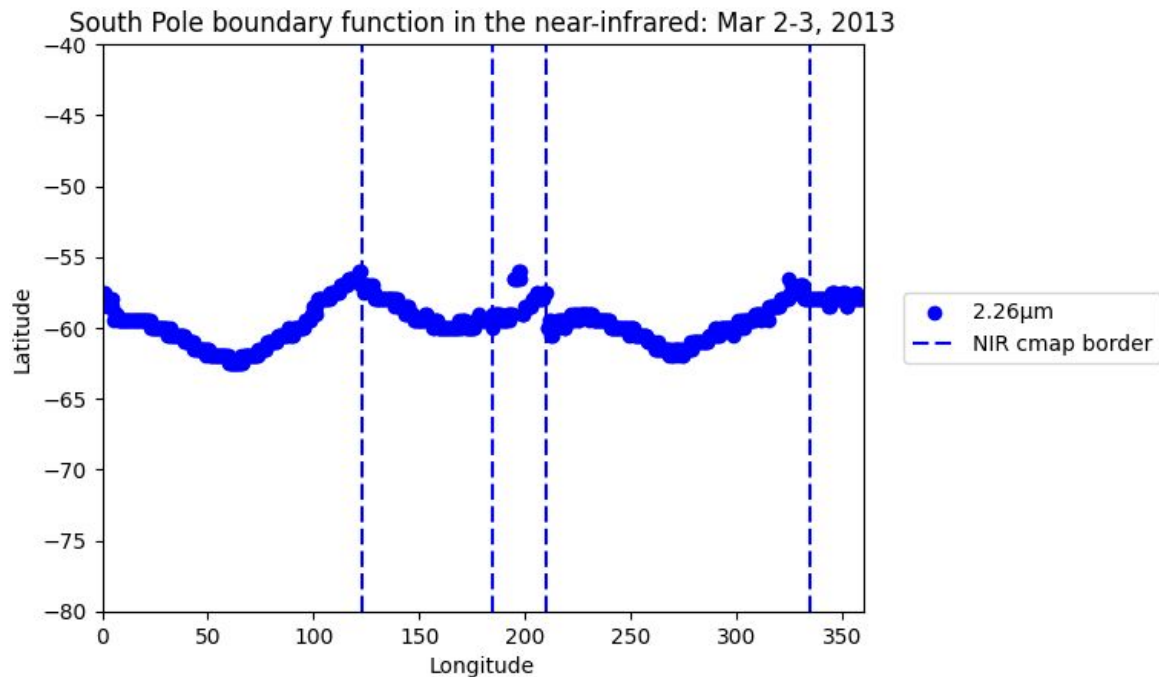
Some Examples

Plotting Jupiter's atmosphere (James Downs)



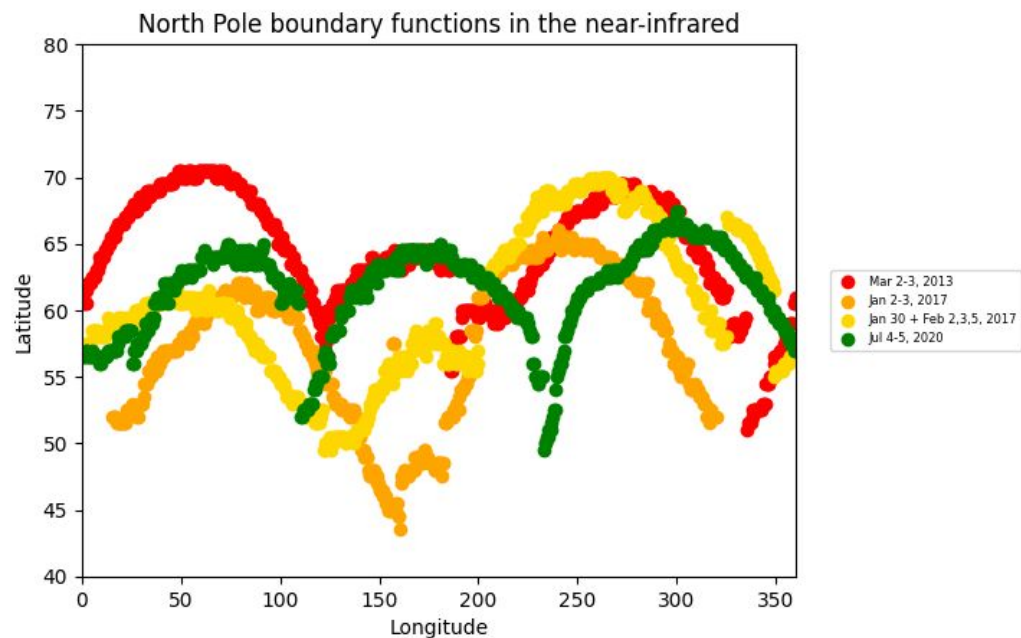
Some Examples

Plotting Jupiter's atmosphere (James Downs)



Some Examples

Plotting Jupiter's atmosphere (James Downs)



(23fa) L16 Exit Ticket Request: 3D Plots

If you're interested in diving in 3D plots, here is the [official link](#) to the 3D plot API, and the [official tutorial](#)

Note: *"3D plotting in Matplotlib is still not as mature as the 2D case. Please report any functions that do not behave as expected as a bug. In addition, help and patches would be greatly appreciated!"*

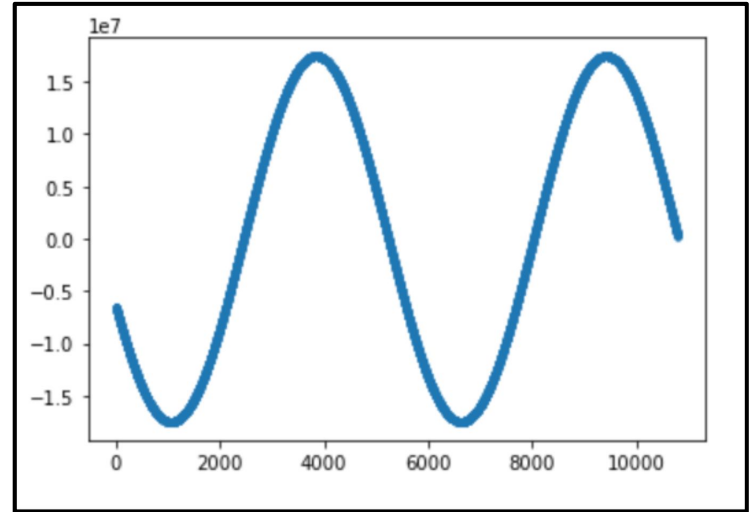
- There are a few other caveats you will see in the official documentation, so be aware of this!

Matplotlib in Student Research

I interned at NASA during the summer of 2021 on the SAGE III/ISS mission through the NASA Langley Research Center. SAGE III/ISS stands for the third-generation Stratospheric Aerosol and Gas experiment, which is located on the ELC-4 dock of the International Space Station and takes data on the gases/ozonewater vapor in Earth's stratosphere. SAGE III takes measurements through occultation, where every time the sun/moon rises and sets, the instrument uses the light that passes through the atmosphere to take data. Through my internship, I developed a software interface to one of the (many) SAGE III/ISS flight telemetry databases using Python and SQL (and Pandas, Numpy, and MatPlotLib as supplements to the main languages). The interface I developed uses a bunch of functions to build and execute queries based on user input for the requested piece of telemetry, requested start time, and requested end time. At the end of the build/execute/decode process, the final data was plotted using MatPlotLib.

Matplotlib in Student Research

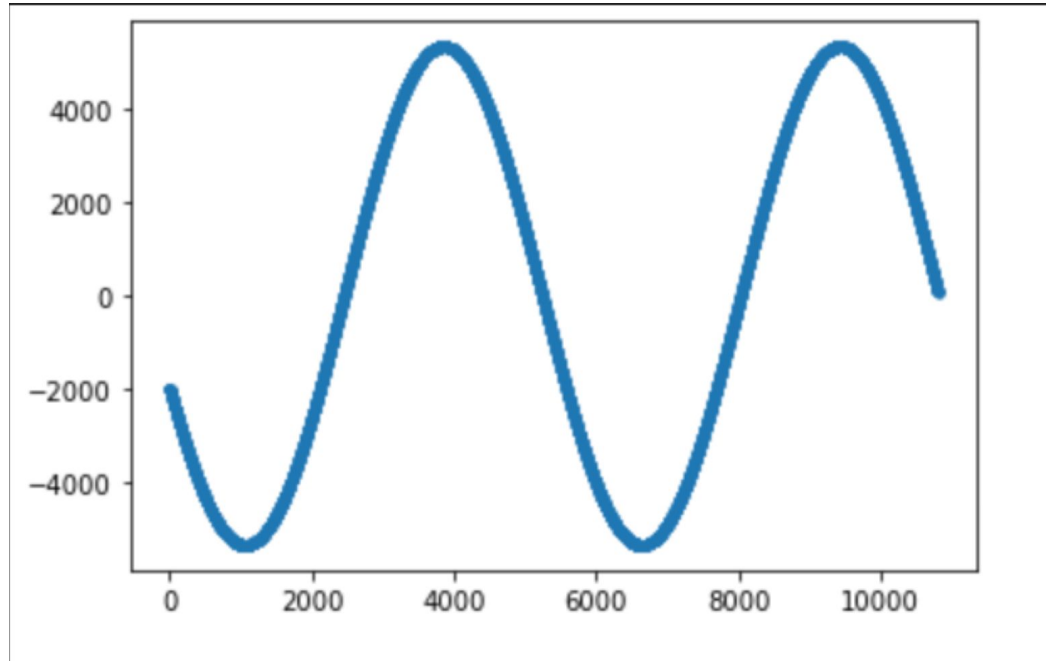
One of the main telemetry pieces that I tested on was the ISS positions. This image represents the ISS Z position.



Note: This is a great application in research; El noted a reminder to label the axis/include a plot title for your own plots :)

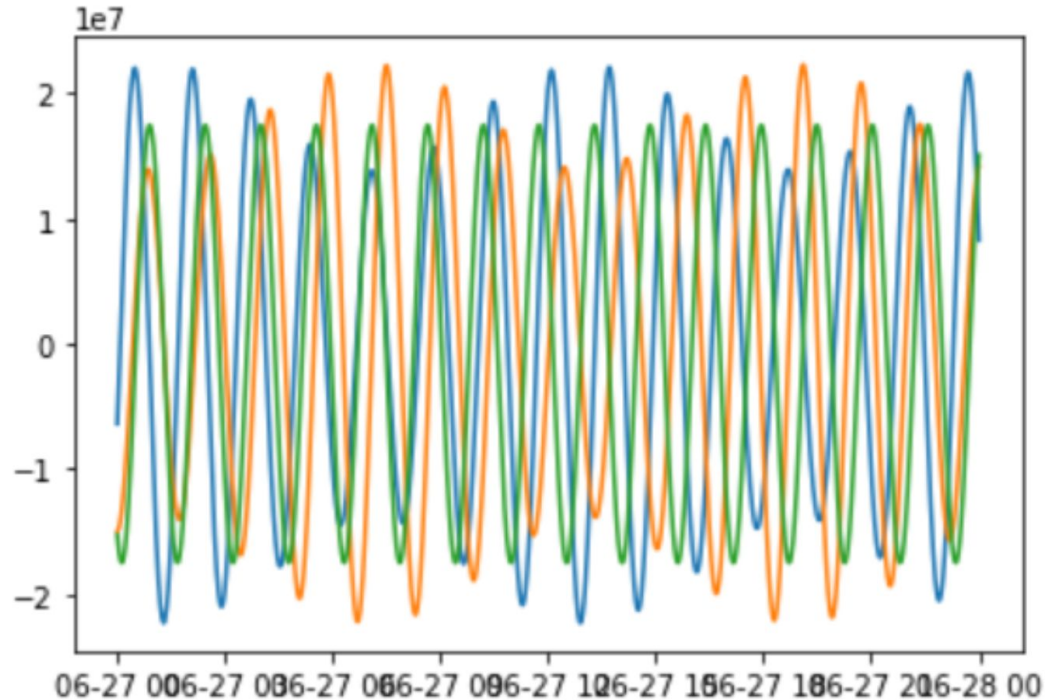
Matplotlib in Student Research

I also coded an option for the user to convert the data and re-plot it.



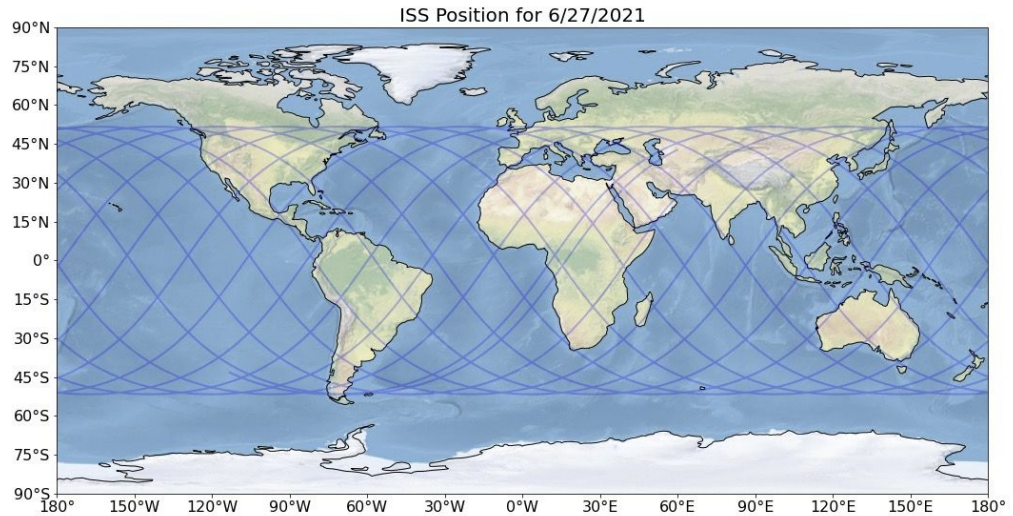
Matplotlib in Student Research

This plot is the ISS x, y, and z positions. Blue is x, yellow is y, and green is z.



Matplotlib in Student Research

And here is the final result! I was able to use the ISS x, y, and z positions to calculate and plot the location of the ISS onto a map of the world.

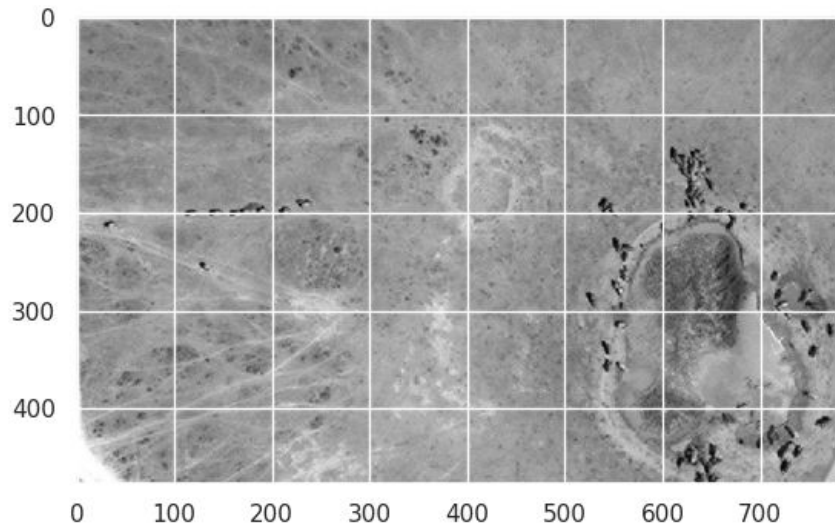
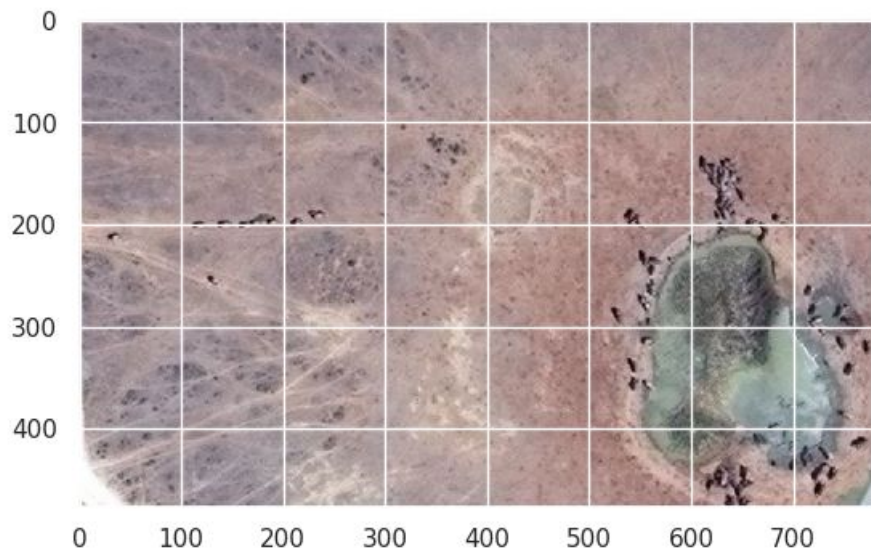


Matplotlib in Future Caltech Classes

Hello everyone! This is Sub, and I just wanted to show y'all how what you're learning now is directly applicable to future classes you guys will be taking. A good chunk of you will be taking Bi1 in the spring, and on one of the sets this past year, we used python and matplotlib to do some image processing on satellite images of the Serengeti to model migration patterns. All of this code was written using concepts learned in CS1 and lots of poring over matplotlib documentation. Take a look!

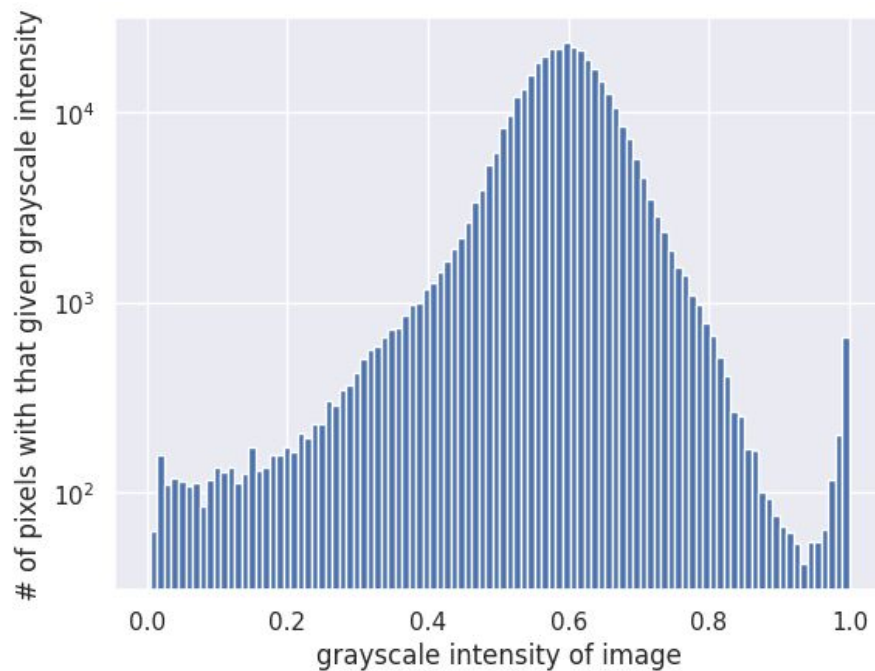
Matplotlib in Future Caltech Classes

We start by looking at the image of the elephants we wish to identify and convert the image to grayscale so it's easier to process.



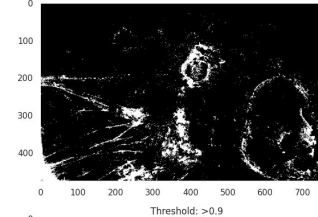
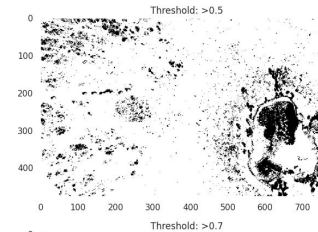
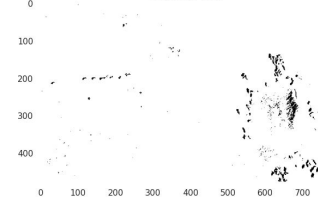
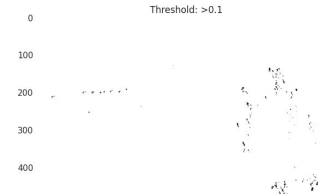
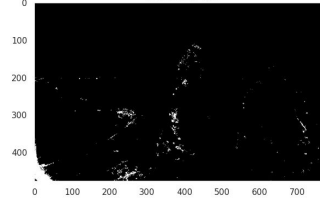
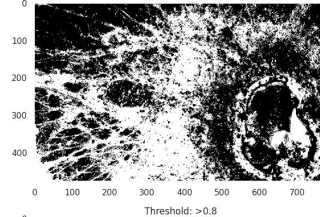
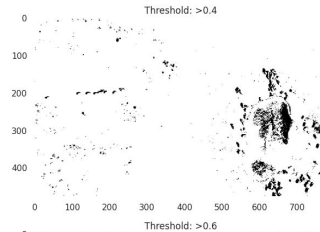
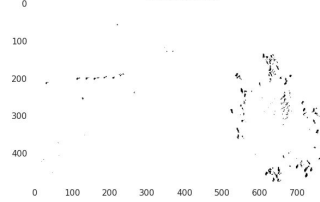
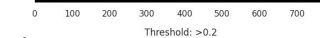
Matplotlib in Future Caltech Classes

After plotting the intensity of all the grayscale pixels using ✨matplotlib✨, we get a better idea of what the threshold needs to be to filter out the much more intense background features.



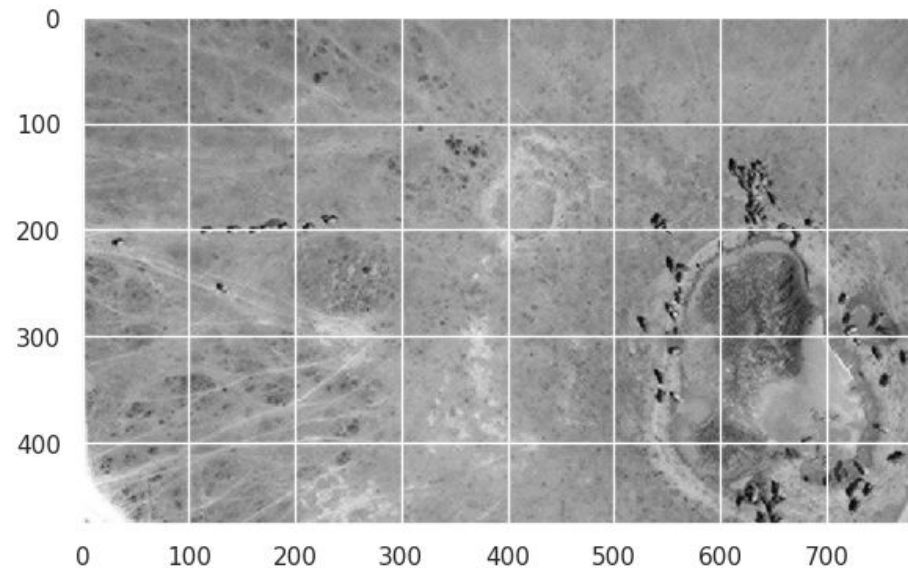
Matplotlib in Future Caltech Classes

We then plot several threshold values to see which values generate the best mask.

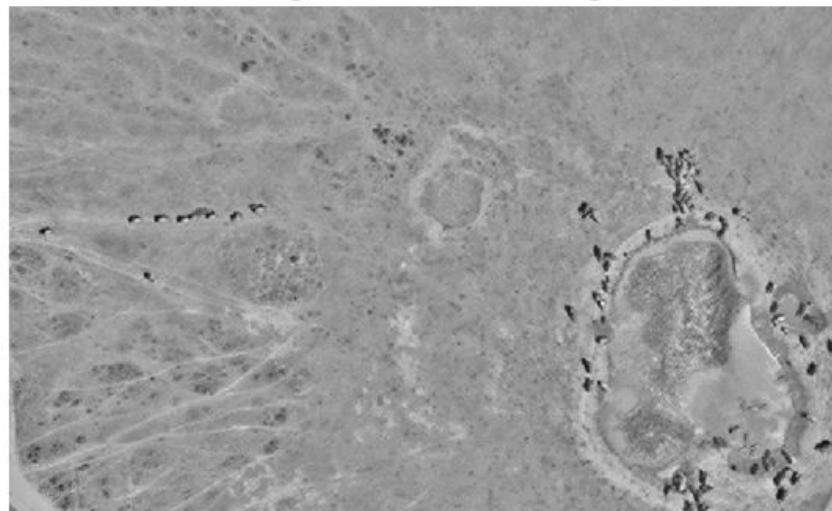


Matplotlib in Future Caltech Classes

Now, using the plotted threshold values and masks, we can subtract the intense background features from the image.

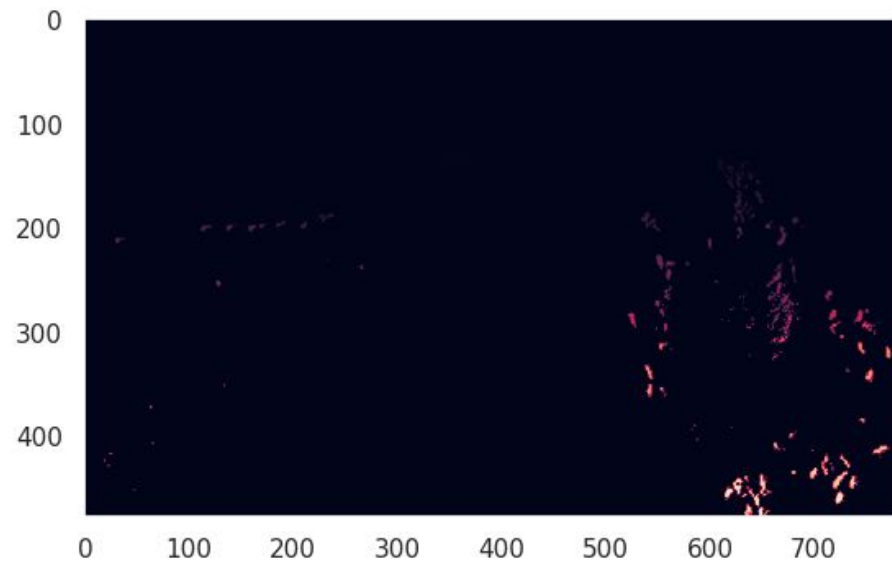
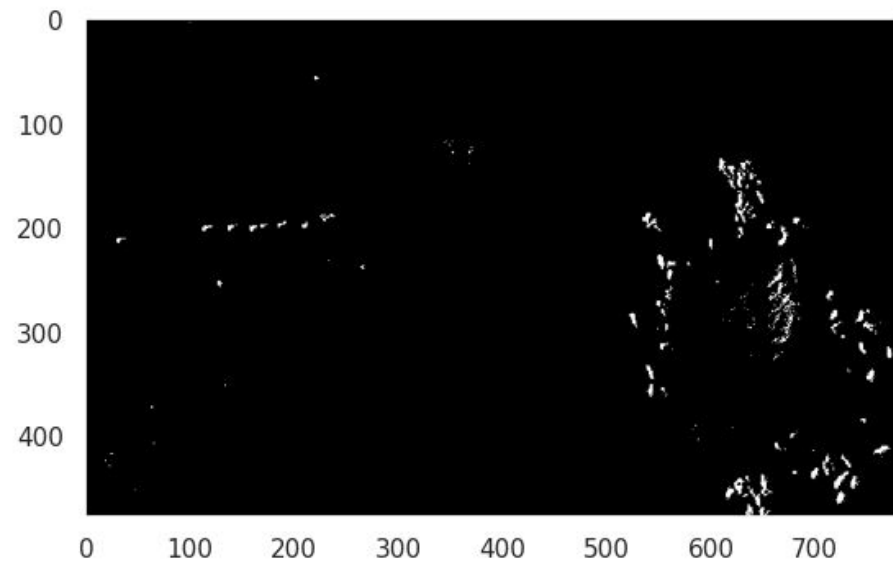


background subtracted image



Matplotlib in Future Caltech Classes

Now, it's just a matter of applying a series of masks to make it easier to identify the elephants!



Matplotlib in Future Caltech Classes

And voila! Using python and matplotlib, we are now able to identify the elephants in the image.

55 elephants identified



MP 5 Part C Showcase

"""

... Credit: Shrishti Kulkarni, 22fa student (current 23fa TA!_

Brief Overview: The program plots two subplots, depicting:

- 1) the most common words used in Taylor Swift's albums,
and their respective frequencies
- 2) the most common words used in each Taylor Swift album, with their frequency
reflected in the size of their marker

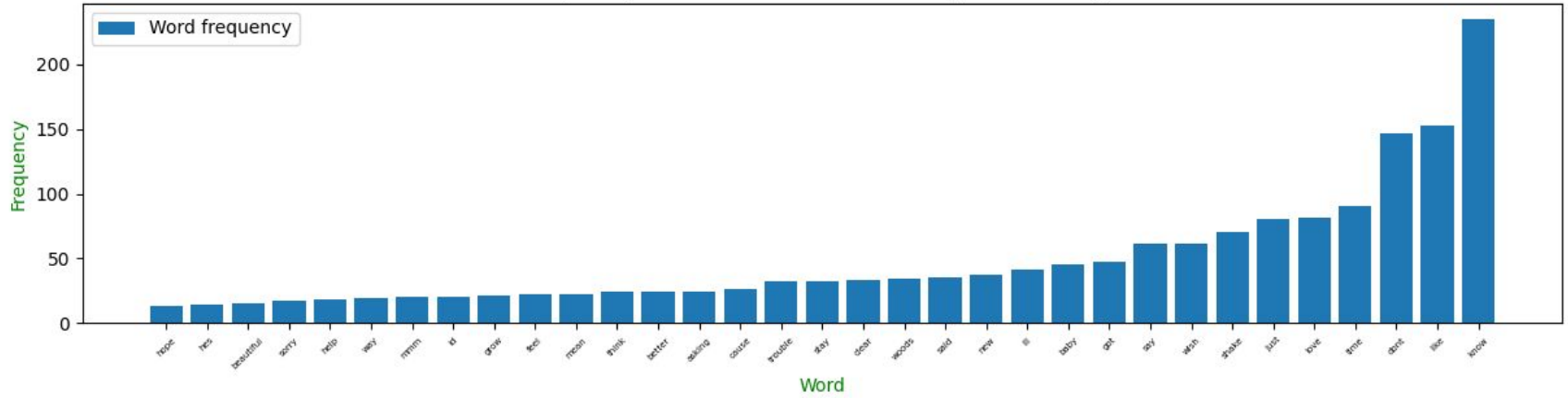
Data Source(s): <https://chart-studio.plotly.com/create/?fid=dalisayd:3#/>

Data Science Question: What are the most frequent words in Taylor Swift albums?

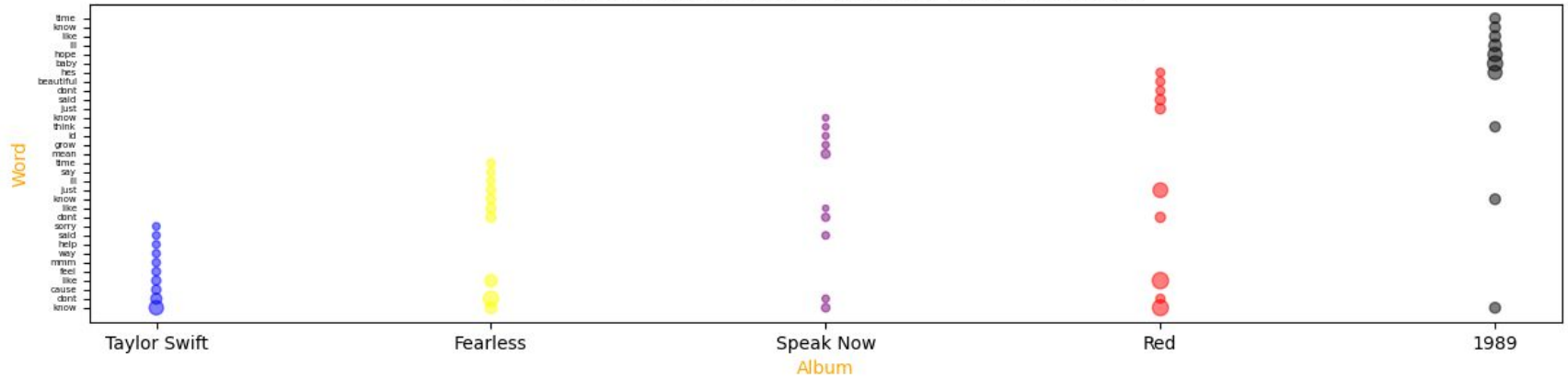
Room for Improvement: Show the marker size when hovering over the marker,
add more annotations and labels

"""

Frequency of most common words in Taylor Swift songs



Frequencies of most common words per Taylor Swift album



Reflection (from MP 5 Showcase Form)

Would you be open to sharing a 2-3 sentence blurb about your inspiration and design/implementation process?

I love Taylor Swift's songwriting, and I wanted to know what the most common words across her albums and in each album, so I decided to look for datasets. When figuring out how to visualize them, I tried to use album cover colors to make it more fun!

Do you have any feedback you'd like to share with EI/the staff about your experience/any suggestions with this new Mini Project? This will be helpful for us in the future!

This time's Mini Project was very accessible and doable, and I really enjoyed Part C. It wasn't too supported (we had to write a lot of code ourselves from scratch), which I thought was extremely helpful.

Do you have any tips for students next year when it comes to learning Matplotlib/exploring features for Part C?

When in doubt, read the documentation, it's super helpful!

Activity: Using Matplotlib to Plot X

Take the next 2-3 minutes to think of an example of a dataset you would like to visualize:

- What is the data of interest?
 - e.g. Lab04 Spotify data, Lab05 sports data, CS 1 22fa student data, pokedex.csv, nobel.csv, planets.csv, etc.
 - "Go Caltech" Sports datasets
- Where might you find the dataset(s)?
 - Course demo data
 - "Go Caltech" website: <https://gocaltech.com/sports/baseball/schedule/2023?grid=true>
 - [AOI datasets](#) (and inspiration for visualizations)
- What is a data science question you would like to explore?

Next Time

Introducing Object-Oriented Programming in Python!