Lab 6

Learning Objectives:

I. **What characters do I type next?** *How do I interact with data in Python and then display this data?*

II. **Let me Google that for you.** *How do I use function 'X'?* This lab will require use of many different functions from libraries. One of the most useful skills a programmer can have is the ability to read documentation and understand how to use certain codes. Thus for this lab we want you to look up documentation on functions and figure out how to use them to help you get the results you want.

While you work is about learning to read documentation and apply it to your code.
Part 1 focuses on scipy.optimize.curve_fit, one of the most commonly used functions in data analysis.
Part 2 consists of a series of exercises in order to better acquaint you with matplotlib.
Part 3 is a short exercise that shows you the power of using plots over calculating things analytically.

While You Work: Google Documentation

It's hard to remember every little detail of a function, so it's useful to be able to look them up. When you need to learn or relearn how to use it up, try Google first. If you figure out that certain search terms work better than others, take note of that for future reference.

Part 0: Get the Lab from Github

Today's lab has starter code. Click on the link https://classroom.github.com/a/ws2IwaN9; the lab is then located at https://github.com/physics91si/lab06-*username*.

Part 1: Playing with scipy.optimize.curve_fit

Open file lab06.ipynb and look at Part 1. Write a function that creates a Gaussian signal for you. (Refer to lab05 for inspiration.) Plot Gaussian functions from x = -5 to 5 with varying widths and heights. Do they look as you'd expect?

Take one of the Gaussians you created and try to fit a line to it using scipy.optimize.curve_fit. *Hint: you already know what the parameters are*. Plot the fit on top of the Gaussian such that both the original function and the fitted function are visible (you can represent one with a solid line and the other with a dashed line, for example). How accurate is the fit?

Now that you're comfortable with Gaussians and how to fit them, load the data (provided for you in the git repository as HD_alpha_data) of the Hydrogen/Deuterium absorption signal. Plot the

data as-is, and then plot the data such that you see an emission signal instead. Now fit a single function to this signal. *Hint: try a double gaussian, which is a simple sum of two different gaussians.*

Plot the emission signal and the resulting fit and make it as "presentation-worthy" as possible. Some things to consider: fiddle with the plot size, include labels, titles, a legend, and play around with different colors.

Part 2: Plotting Exercises

Look at Part 2 of lab06.ipynb. The following exercise should help you become comfortable with the more detail-oriented aspects of plotting, as well as searching up what you don't know.

First generate two random sets of 100 data points, ranged from 0 to 1, and store them in two arrays. With this data, you will create a pedagogical figure:

1. Create a figure with 4 subplots, arranged in a 2x2 format. (They do not have to share the same axis values.) On the first subplot, create a scatter plot using the first 25 values in each array; on the second, use the second 25 values; so on and so forth.
2. Give each set of data points a different color and opacity.
3. Have the left two plots vary their data point sizes so that they increase with increasing y values.
    a. Challenge: Using a colorscale, have the first plot vary its color based on x value.
4. Have the right two plots vary their data point sizes so that they increase with increasing x values.
    a. Challenge: Try to have the fourth plot vary its data point sizes so that they decrease with decreasing x values.
5. Get rid of the visible "tick" marks on each x and y axis.
6. Give each subplot a title, and then give the overall plot a title.
7. Experiment with the image you have just created—try to make it as aesthetically pleasing as possible! Possibilities include: playing around with spacing of the subplots, font size, and overall image size.

When you're done, save the final figure as a pdf-type file in your repository.

Part 3: Plotting electric field lines

Now look at Part 3. Create three different arrays, each containing information for the two point charges: one array containing their x-positions, one containing their y-positions, and one containing their charges. Then create an X array and a Y array, both from -31.5 to 31.5 and containing 64 points each.

With this information, obtain the flux and plot the contours of the flux to see the electric field lines. *Hint: Use plt.contour(X, Y, Z, N) where Z is what you want to make a contour plot of and N is the number of contour lines you want to draw.* You may also want to plot the positions of the

charges to better visualize what is going on.

Make the plot “pretty,” and play around with different numbers. What do you see?

Part 4: Submit Your Lab

In case you forgot -- save your work in the Jupyter notebook, then shut it down and enter the commands (from Terminal/Anaconda Prompt/whichever command line you’re using):

1. `git add -A`
2. `git commit -m "<insert commit message here>"`
3. `git push`