Lab 8*

These exercises will use many ideas from the previous sessions. Our goals are to:

- Develop basic graphing skills
- Import a data set
- Perform a simple fit of a model to data

Example 1: HIV Viral Load

Here we explore a model of the **viral load** -the number of virions in the blood of a patient infected with HIV- after the administration of an antiretroviral drug. One model for the viral load predicts that the concentration V(t) of HIV in the blood at time t after the start of treatment will be:

$$V(t) = Ae^{+\alpha t} + Be^{-\beta t} \quad (1)$$

The four parameters A, α, B, and β are constants that control the behavior of the model. A and B specify the initial viral load, α is the rate at which new cells are infected, and β is the rate at which virions are removed from the blood. In this lab, you will use what you have learned about Python up to this point to generate plots based on the model, import and plot experimental data, and then fit model parameters to the data.

1-1 Explore the model:

To get started, launch Spyder, import NumPy and PyPlot, and then create an array “time” for the time intervals between 0 and 10 seconds with 101 numbers.

Next, you will evaluate a compound expression involving this array by using the solution of the viral load model given in Equation 1.

* Adapted from “A Student’s guide to Python for Physical Modeling” by Jesse Kinder and Philip Nelson.

The first step is to give the constants in the mathematical equation names you can type, such as alpha and beta instead of α and β. It is wise to give longer, more descriptive names even to the variables whose names you can type: for example, time fort and viral load for V (t). Now, set B = 0, and choose some interesting values for A, α, and β. Then, evaluate V(t).

you should now have two arrays of the same length, time and viral_load, so plot them. Create a few more plots using different values of the four model parameters.

1-2 Fit Experimental Data:

Now let's have a look at some experimental data.
Follow the instructions from Lab 7 to obtain the dataset “01HIVseries”. Copy files README.txt and HIVSeries.csv into your working directory, or be sure you can locate these files using paths. The HIVseries files contain time series data. Read the fie README. txt for details.

Import the data set into Python by reading HIvseries.cvs into an array. You are free to give the data whatever name you like, but we will refer to the array as hiv_data.
Find the variable you created in the Variable Explorer. It has two columns of data. The first column is the time in days since treatment of an HIV-positive patient; the second is the concentration of virus in that patient's blood, in arbitrary units.

Next, we are going to visualize the data. In order to plot the viral load as a function of time, you will need to separate the data into two arrays to pass to plt.plot. Do that and plot the data points now.
Don't join the points by line segments. Instead, make each point a marker (symbol), for example, a small circle or plus sign. Label the axes of your plot. Give it a descriptive title, too. To review such embellishments refer to the documentation of matplotlib.pyplot.plot.

What to turn in (submit on GitHub):

A. Plot the experimental data points and the function in Equation 1 on the same axes. Adjust the four parameters of the model until you can see both the data and the model in your plot. (Remark: You probably know about black-box software packages that do such "curve fitting" automatically.In this lab, you should do it manually, to see how the curves respond to changes in the parameters.) The goal is now to tune the four parameters of Equation 1 until the model agrees with the data. It is hard to find the right needle in a four-dimensional haystack! We need a more systematic approach than just guessing.

   Consider the following: Assuming $\beta > \alpha$, how does the trial solution behave at long times?
   If the data also behave that way, can we use the long-time behavior to determine two of the four unknown constants, then hold them fixed while adjusting the other two?

   Even two constants is a lot to adjust by hand, so let's think some more: How does the initial value V0) depend on the four constant parameters? Can you choose these constants in a way that always gives the correct long-time behavior and initial value?

B. Carry out this analysis so that you have only one remaining free parameter, which you can adjust fairly easily. Adjust this parameter until you like what you see.

C. The latency period of HIV is about ten years, or 3,600 days. Based on your results, how does the inverse of the I-cell infection rate, 1/a, compare to the latency period? Is the latency period long because it takes HIV a long time to infect new cells? Or does the model suggest another scenario?

Example 2: Bacteria Growth

Now we turn our attention to genetic switching in bacteria. In 1957, A. Novick and M. Weiner studied the production of a protein called beta galactosidase in E. coli bacteria after introducing an inducer molecule called TMG.

2-1 Explore the model:

Here are two families of functions that come up in the analysis of the Novick-Weiner experiment:

$$V(t) = 1 - e^{-t/\tau} \qquad \text{and} \qquad W(t) = A(e^{-t/\tau} - 1 + \frac{t}{\tau}) \qquad (2)$$

The parameters τ and A are constants.

What to turn in (submit on GitHub):

A. Choose A = 1, τ = 1, and plot W(t) for 0 < t < 2.
B. Make several arrays W1, W2, w3, and so on, using different values of τ and A, and plot them all on the same axes.
C. Change the colors and line styles (solid, dashed, and so on) of the lines.
D. Add a legend to help a reader sort out the curves. Explore some of the other graph options.

2-2 Fit experimental data

Follow the instructions of Lab 7 to obtain the data set “15novick”. Copy “g149novickA.csv”, “g149novickB.csv”, and README. txt into your working directory, or be sure you can locate these files using paths. The files contain time series data from a bacterial population in a culture.

What to turn in (submit on GitHub):

[Remark: Again, you are asked not to use an automatic curve-fitting system. The method suggested earlier and the hints below will give you a deeper understanding of the math than accepting whatever a black box spits out.]

A. Plot the experimental data points and your trial functions for V (t) (see Equation 2) on the same axes, as you did for W(t) before.

   When you plot the experimental data, do not join the points by line segments; make each point a symbol such as a small circle or plus sign. Label the axes of your plot. Select some reasonable values for the parameter in the model, and see if you can get a curve that fits the data well. Label the curves, then add a legend to identify which curve is which. [Hint: To find a good estimate of τ, make a semilog plot of the quantity “1.0 - data” versus time, where “data” is the array of experimental data points. (Can you explain why this is helpful?

B. Now try the same thing using the data in “g149novickB.csv”. This time throw away all the data with time value greater than ten hours, and attempt to fit the remaining data to the family of functions W(t) in Equation 2.
   [Hints: (i) You can "throw away data" by slicing an array.
   (ii) At large values of t (but smaller than ten hours), both the data and the function W(t) become straight lines. Find the slope and the y intercept of the straight line determined by Equation 2 in terms of the two unknown quantities A and τ. Next, estimate the slope and y intercept of the straight line determined by the data. From this, figure out some good initial guesses for the values of A and τ. Then, tweak the values to get a nicer looking fit]