**Instruction Manual and Tutorial**

**Introduction**

This program accompanies the paper (citation will be available upon publication). It is designed to draw control charts for ELISA data. Contact information can be found in the paper or through the Github. This has been cross-posted on Scholar one (University of Cincinnati). Feel welcome to edit and adapt the code as needed. Report any issues or bugs on the Github page and they will be addressed as soon as possible.

**Pre-Requisites**:

Python 2

Required Python Modules:

numpy

matplotlib

Tkinter

Glob

I recommend installing all of this (Python and packages) through Anaconda. This will ensure that your Python is set up correctly and you will have all the packages you need.

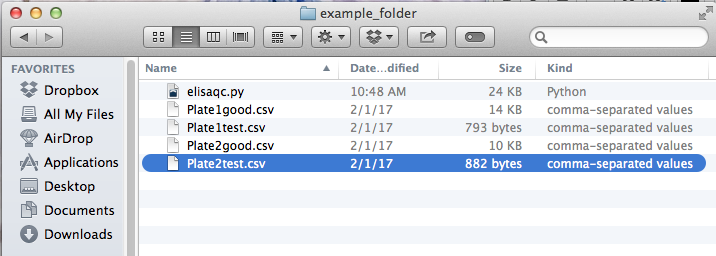
Link:

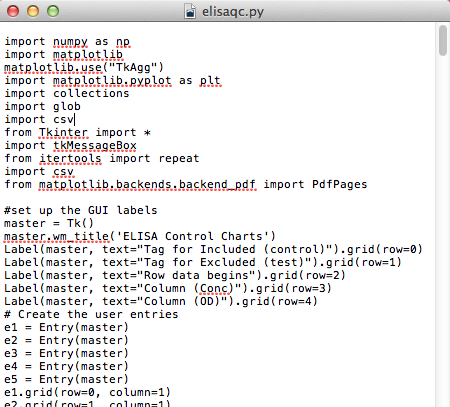
https://www.continuum.io/downloads

**Limitations/requirements**

In order to avoid code alterations, you must have used 8 concentrations, in triplicate. Data and files must also be formatted according to the specifications included below.

**Installing and Opening**

First, download the folder from the internet and put it somewhere your computer. For this example I have placed it in a folder called example\_folder, located on my desktop. Copy the files you wish to analyze into the same folder. To be clear, the file containing the script must be in the same folder as the csv files containing the data. My folder now looks like this, containing the .py file and the 4 .csv files containing my 4 curves I wish to analyze. 

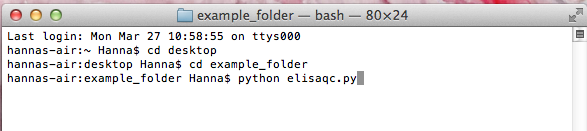
Make sure that the .py file opens as a text file. Do this by opening it once using notepad (Windows) or text edit (mac). The file should look like this. I recommend setting it so all files of this type open using this program. 

Next, you must launch the GUI. Open the command line (windows) or Terminal (mac) and navigate to the folder where you stored your files.

In mac, this can be done by hitting command+space then typing “terminal”

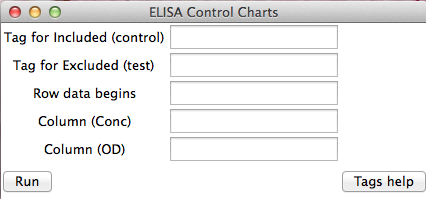
On a PC, this can be done by searching “cmd” on your start menu and selecting the command line.

Once open, navigate to where you stored the files. Do this by typing “cd” for change directory, followed by the path to your folder. Once in the correct folder, call the program by typing “python elisaqc.py”. Here is an example of the line of commands you should enter (hitting enter after each command):



The names of the folders cannot have certain characters in them or they will confuse the command line. White space (spaces, tabs, returns) are particularly problematic, so use an underscore instead of a space if such is necessary.

When you press enter, it should call up the interface, which should look like this:



**Preparing your data:**

If you wish to use the program without altering the code (instructions available later), your data must be formatted in a very specific manner. We have attempted to make this format similar to that which is automatically exported. Below are a list of requirements:

File naming requirements:

1. The file name must end in .csv, and be a .csv type file.
2. A ‘tag’ must be entered for files which you wish to include in the mean and standard deviation calculations to differentiate them from the ones you wish only to graph. This tag must be entered exactly the same (case sensitive) at the end of all the file names, immediately prior to the .csv. The same must be done for the files you wish only to display.

For example, if I have two ‘good’ plates I want to be included in the calculations, and another two ‘test’ plates I wish only to display, I could have my file names like this:

Plate1good.csv

Plate2good.csv

Plate1test.csv

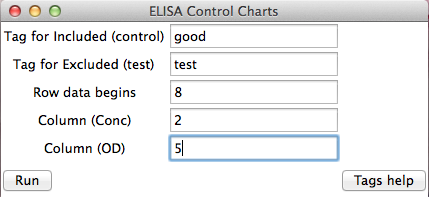
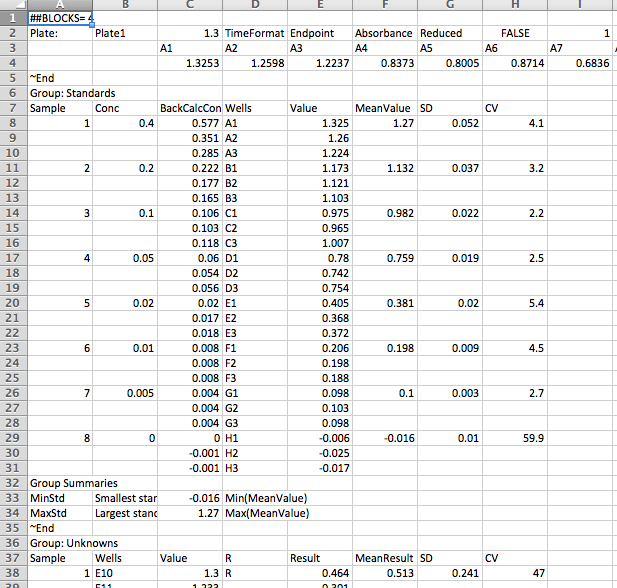
Plate2test.csv

If you do not have a test group, and want to only analyze one group of data, simply enter “none” into the test group line.

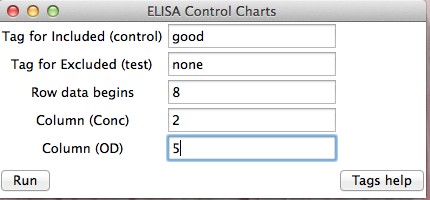
File formatting requirements:

1. Most importantly, only one standard curve can be included per plate. Any extra will not be included in the analysis. If you are routinely running more than one standard curve per plate, I have written a quick script that splits .xlsx files and creates three new files, one with each curve. This script does not contain a GUI, so you will need to modify the code in order to customize it. The code contains notes on where to input your file names and row values. There are multiple free programs online to convert xlsx files to csv files.
2. You may not have blank rows at the top of your file.
3. Data must be in triplicate. Code alterations are needed to accommodate other replicates if needed, instructions later.
4. You must have 8 concentrations on your curve. Again, code alterations are available to accommodate other formats.
5. There may be other data below the standard curve (samples for example), or values in other rows. This is fine, all these values will simply be ignored.

An example of a properly formatted file and the corresponding GUI entry for the folder shown previously would look like this:

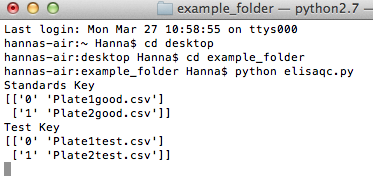


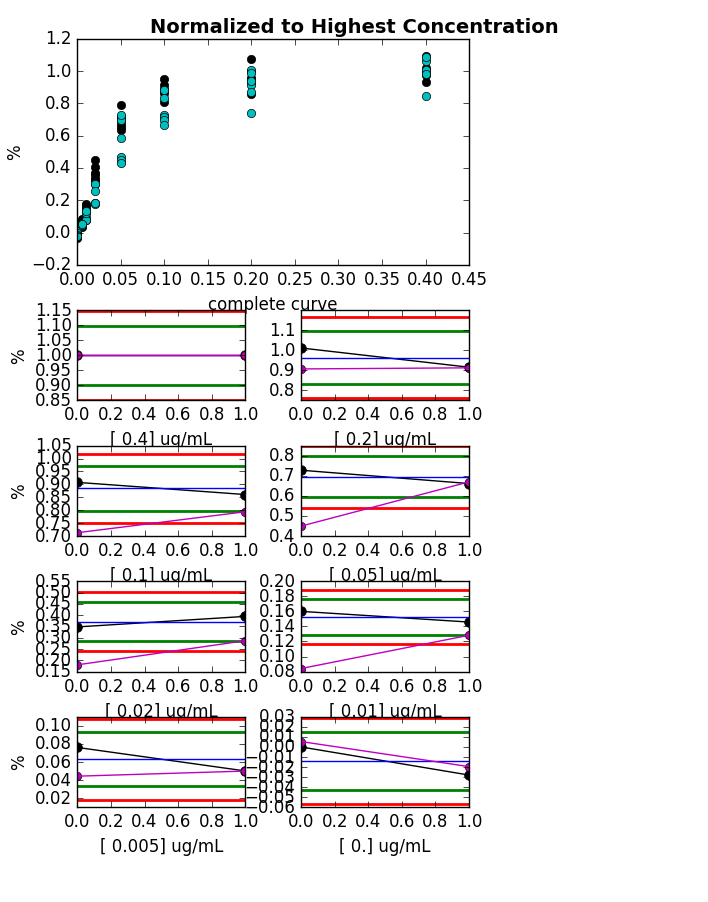
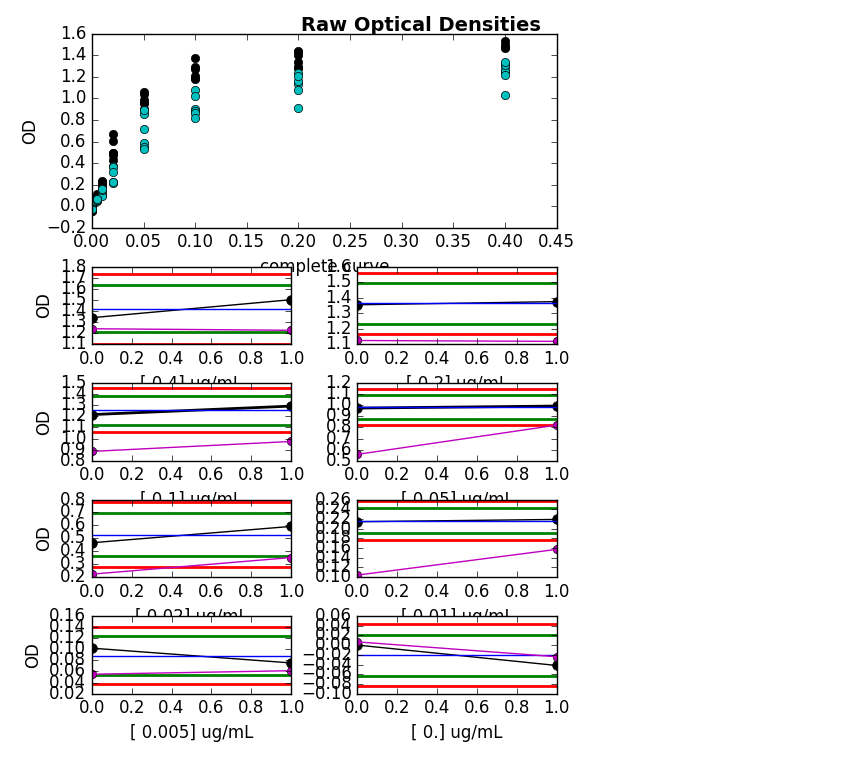
If we only wanted to look at one group of data (the control group for example) the GUI would look like this:



**Output**

The program should produce a graph that will both pop-up and be saved to your folder. A key will be displayed in your terminal showing what number corresponds to each file you entered. A file named “ELISASTATS.csv” will also be generated containing statistics describing your data. If any values fall outside of 2 or 3 standard deviations, this will also display what file they were in and at what concentrations. Example graphs and keys for the example above are shown:

****

****

**Altering the Code**

Obviously, changing the code provides more flexibility, but at the risk of the code not functioning. These are a few suggestions only for what we anticipate to be common problems. However, feel free to alter the code as desired if other issues/applications arise.

1. Your file names do not end in .csv

This solution is actually fairly easy. The lines identifying which files to use are lines X and X. Simply replace the text in quotations to what your file name actually ends in. For example, the line currently reads

“for files in glob.glob("\*%s.csv"%(userinput1)) :”

If, for example, all my files ended instead in “ending”, I would alter the line to read

“for files in glob.glob("\*ending:”)

This will, or course, override the GUI entry, so if you take this option, ignore those inputs.

1. You did not run your curve in triplicate

This is a little more complicated, but still quite doable. Within the loops, you will find that the data is cut into 8 , 3 value segments (8 concentrations, 3 values/concentration). For example, line X reads

“array1=array[0:3]”

and this is repeated 8 times, for arrays 1-8. To change this, simply change the numbers accordingly, for example, for duplicates,

“array1=array[0:2s]”

and repeat this for all the concentrations.

You must change this according to your needs.

1. You did not use 8 concentrations.

This is potentially the most difficult alteration. Essentially, you must remove/add all the extra concentrations that you need. This must be done throughout the whole script, including both loops and the graphs.

**Issues and requests**

I would like this code to be as usable and accessible as possible. If you encounter any problems, please report them on the Github Issues page. If you have any requests for making the code more amenable to your data, feel free to contact me or post an issue and I will do my best to accommodate. It is important to me that this program help as many researchers as possible, so don’t hesitate to ask for anything that will help you achieve your needs.