

Gluing Galaxies: A Jupyter Notebook and Glue Tutorial

Abstract:

This tutorial will be utilizing an app called “Jupyter Notebook” to create and visualize scatter plots using python code. The data we want to plot, which will be explained more in-depth in the introduction, is meant to show different ways that we can measure galaxy ages. One plot will be a “color-color plot”, which gives the relative *flux* (or “brightness”) ratio of ensembles of stars within a galaxy at two different wavelengths (and their colors as a result), while the other will show two spectral age diagnostics, D4000 and $H\delta$ (pronounced “H-Delta”), plotted against each other. We will then use another app called “Glue” to plug in the new plots we’ve made and see how the colors of galaxies correlate with their spectral age diagnostics.

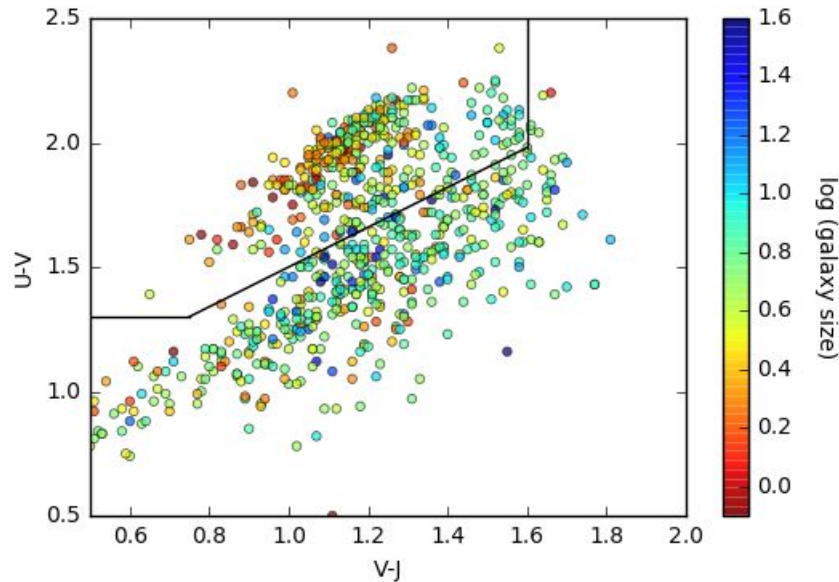
Introduction:

When it comes to astronomy, there’s quite a bit of jargon that can make it seem pretty daunting. This introduction is meant to help you understand what we’ll be doing as well as provide some helpful terms. For starters, the galaxies we’ll be analyzing are called *quiescent* galaxies, which are galaxies that have stopped forming stars (think “quiet”). Our own Milky Way is actually somewhere in between in that it’s still forming stars here and there, but isn’t making enough at the rate to be considered a “star-forming galaxy”.

The sample of data used in this tutorial is called the Large Early Galaxy Astrophysics Census, or LEGA-C, and contains data of several thousand galaxies from redshift 0.6 to 1. More information on this, as well as specific details on the data the team has collected, can be found on their [homepage](#). As for what *redshift* is, it can be a little easier to understand if you know how the Doppler Effect works. But, to simplify: the light that travels out from galaxies gets stretched since the universe, and space itself, is expanding. The wavelength of this shifted light then stretches toward the red part/lower frequency end of the visible light spectrum causing more distant galaxies to have a reddish tinge. In other words, galaxies that are further away and existed much earlier in the cosmic timeline will have a greater redshift.

**As a bonus, [this](#) calculator and [this](#) calculator can help give you an idea of what redshift is in terms of other measurements.

There are many, many different ways the LEGA-C data has been categorized. If you skip to the end of the “Getting Started” section, you can view the data we’ll be using for the tutorial. There are headings such as `galfit_re_kpc` and `fast_lmass`, but the ones we’re going to focus on are `UV`, `VJ`, `D4000_N`, and `LICK_HD_A` ($H\delta$). The data in this file represents the quiescent galaxies from the entire catalog. Here’s what the rest of the data looks like plotted by U-V vs. V-J color-coded in terms of galaxy size:



The U, V, and J filters represent wavelengths of light filtered out to meet specific requirements. To compare the relative amount of light emitted in each of the respective filters, the U-V color has one filter more toward the blue end of the visible light spectrum and another toward the red end, while the V-J color shows the visible light relative to the infrared light. Together, these two diagnostics can phase out quiescent/redder galaxies, which are enclosed in the upper left corner of the plot.

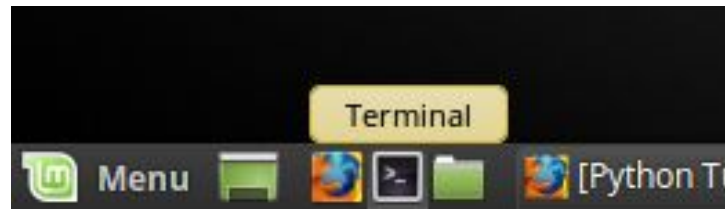
Another important concept when it comes to understanding galaxies is their age. However, age is a bit difficult to define in terms of galaxies. Galaxies can be considered billions of years old if you're talking about when their first stars formed, which was a little after the Big Bang. Others may say their real age comes into play when the galaxy has formed half its *stellar mass*, which is the total mass of all stars in the galaxy. A blog post from the CANDELS Collaboration site titled "How Old are Galaxies?" puts it best when they write, "Early galaxies started forming their first stars when they had acquired only a tiny fraction of their present day material. They subsequently grew larger as more gas fell in at later times, and when they merged with other galaxies." Therefore, it makes more sense to measure a galaxy's (peak) star formation rate and use that to help determine its age

This can all be drawn back to understanding H δ absorption lines and the D4000 Angstrom break. H δ represents the wavelength measured by the photons that hydrogen particles absorb while D4000 represents the strength of the *4000 Angstrom* break (400 nm), which is just many different *metal* (any element that's not hydrogen or helium) absorption features placed on top of one another. When these two measurements are plotted against each other, they can help us find the age of a galaxy. A stronger D4000 means an older galaxy since the emitted light is dominated by lower mass, longer lived stars that have the accumulation of metal absorption lines from their stellar atmospheres. On the other hand, H δ lines are strongest in galaxies that have an age of one *gigayear* (gyr for short, one billion years) because the stars with similar lifespans have the strongest absorption features associated with the hydrogen *Balmer Series* atomic transitions (when an electron goes from the n=2 to n=6 state it absorbs a photon with an energy corresponding to a wavelength of 410.2 nm). When the two are plotted against each other, it will make most sense to see an inverse relationship between them.

Getting Started:

All of the programs that are being run are done so through Linux Mint, which I installed through Windows 8 using [this](#) link. You can find many tutorials online detailing how to dual boot Linux with your computer, although it's not required that you do so (or even use linux at all, it just makes things easier for us).

Once you've downloaded Linux, it should (for the most part) look like this, with an icon linking you to the terminal in the bottom left corner as shown here:



Before beginning this tutorial, you'll want to [download](#) Anaconda through their website. Whichever operating system you choose, make sure you choose the "Python 2.7" version. This way, any commands you type in later will show up correctly in Jupyter. Any other steps you need to take to make sure it's been downloaded properly will be found there, although you must open a new terminal window and run `conda` to verify your terminal system can find it.

This next part is a summary of the tutorial found [here](#). If you wish to learn more, feel free to read the documents found there.

Assuming you are using Linux, go ahead and type the following into the command line of the terminal:

```
$ conda config --add channels http://ssb.stsci.edu/astroconda
```

This allows anaconda to install packages directly from Astroconda. Once this is complete, you can download the `stsci` metapackage, which includes Glue and Jupyter Notebook, by typing this into the command line:

```
$ conda create -n astroconda stsci
```

Once this is done, type in the following command to activate astroconda and give yourself the ability to access all the packages you need:

```
$ source activate astroconda
```

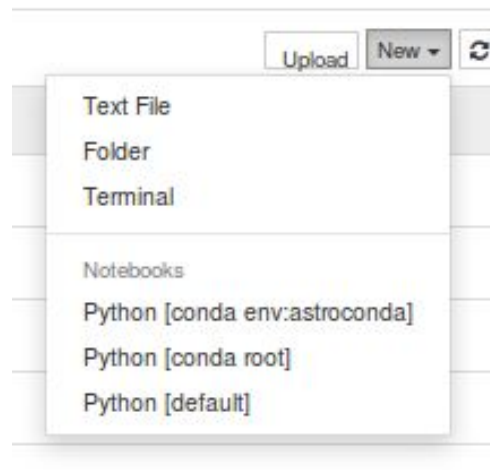
You won't need to run this command each time you boot up Linux or anything like that. This is just to help make sure that everything is running properly.

Lastly, you'll want to create a new directory to store the files we're going to use for this tutorial. I called mine "legac_files", but you can call it whatever you'd like. Just make sure you adjust the second input definition for `catalog_path` accordingly. Then, download the catalogs using [this](#) link.


Now that you've got the materials, let's get into the nitty-gritty...

Jupyter Notebook

The first thing you'll need to do is type `jupyter notebook` into your command line to open your notebook. After this, you will see a dropdown box in the upper right hand corner that says "new", which will look something like this when you click it:



Go ahead and select [conda root] to start. Once you do so you will be brought to a page that allows you to input code. You can name and save the notebook as whatever you like. Once you're here, import some utilities that will allow you to use math functions as well as visualize your data.

You can run this code by pressing the  button. Don't forget to save as you go, as Jupyter doesn't do a very good job saving your progress.

```
In [1]: from astropy.io import ascii
        from numpy import *
        import matplotlib.pyplot as plt
        %matplotlib inline
```

Make a variable that points to the directory holding the catalogs. I used "rochelle" since that's my name, but you should name it whatever that directory is called for you. This step also assumes you stored the `legac.cat` file in a directory called "legac_files".

```
In [2]: catalog_path="/home/rochelle/legac_files/"
```

Read in the "Legac" Catalog

First you'll want to read in the data. Since this is an *ascii file*, which is a text file including *ascii* characters (code that associates 0's and 1's for each symbol in a character set, such as letters, digits, punctuation marks, etc.), we're going to do this like so:

```
In [3]: data = ascii.read(catalog_path+'legac.cat')
```

Select Galaxies

Since we'll be using U-V and V-J specifically, you'll want to make arrays that are linked directly to that data.

```
In [4]: UV = data['UV']
        VJ = data['VJ']
```

Once we've done this, we want to make a selection of the quiescent galaxies. Way back in the introduction you saw a color-color diagram with a section blocked out. This part right here is meant to select the galaxies within that section, aka our quiescent galaxies. To fully understand why we're using these numbers for our selection, head to [this](#) link. On page two, you'll see a plot that looks almost identical to our original one. As you can see, the points must be greater than 1.3 for U-V and less than 1.5 for V-J; " $UV > 0.8 * VJ + 0.7$ " is the formula of the sloped line; and the "`(data['use'] == 1)`" just means we want to use galaxies that are of good quality as opposed to anything that got through but wasn't considered good enough data, such as stars.

```
In [5]: quiescent_selection = [(UV > 0.8*VJ+0.7) & (UV > 1.3) & (VJ < 1.5) & (data['use'] == 1)]
```

Next, we take the data from U-V and V-J and select out the quiescent galaxies from those data points.

```
In [6]: uv_quiescent = UV[quiescent_selection]
        vj_quiescent = VJ[quiescent_selection]
```

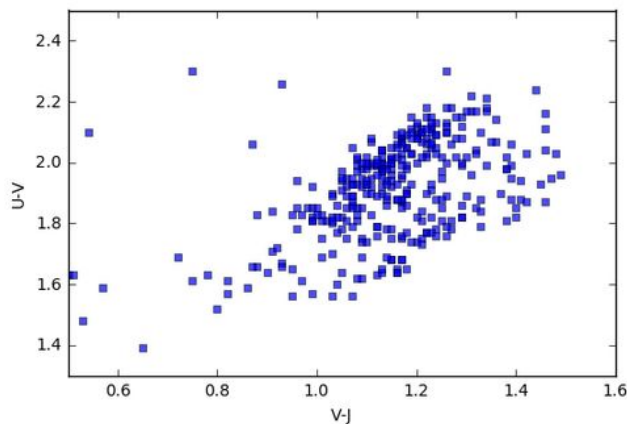
V-J vs. U-V

With all the pieces in place, let's build our scatter plot. Make the quiescent selection of the V-J filter your x-axis and the quiescent selection of the U-V filter your y-axis. The marker is set to 's' for square (the entire list of markers can be found [here](#)) and the size of the marker is set to 16. Alpha adds transparency to the points and linewidth is just the width of the lines around the points. There's no need for the marker, size, transparency, or line width to be set in stone, so play around with the code a bit and see what happens. The x (`plt.xlim`) and y (`plt.ylim`) bounds are then determined from the plot in the **select galaxies** section and we want our x and y axes to have labels so we'll call them V-J and U-V respectively.

```
In [7]: plt.scatter(vj_quiescent, uv_quiescent, marker='s', s=16, alpha=0.7, linewidth=0.5)
        plt.xlim(0.5, 1.6)
        plt.ylim(1.3, 2.5)
        plt.xlabel('V-J')
        plt.ylabel('U-V')
```

If you've done everything correctly, this is what you should see when you run this block of code:

```
Out[7]: <matplotlib.text.Text at 0x7f6806f6a450>
```



Tweak the Data

Congrats, you just made your first scatter plot in Jupyter Notebook! It all looks pretty good, but we still need to make some finishing touches.

These arrays are going to be used for the H δ vs. D4000 plot as well, so the name is done out of convenience. Take data points from H δ and D4000 that give quiescent galaxies and have a value greater than -99. Galaxies that don't have a measured value for either parameter are given this value of -99 so we want to make sure to exclude those.

```
In [8]: HDelta_D4000_selection = [(data['LICK_HD_A'][quiescent_selection] > -99) &
    (data['D4000_N'][quiescent_selection] > -99)]
```

You can copy and paste the code from block 7 into the next line. Change it to look like this.

```
In [9]: plt.scatter(vj_quiescent[HDelta_D4000_selection],uv_quiescent[HDelta_D4000_selection],marker='s',s=16,
    alpha=0.7,linewidth=0.5)
plt.xlim(0.5,1.6)
plt.ylim(1.3,2.5)
plt.xlabel('V-J')
plt.ylabel('U-V')
```

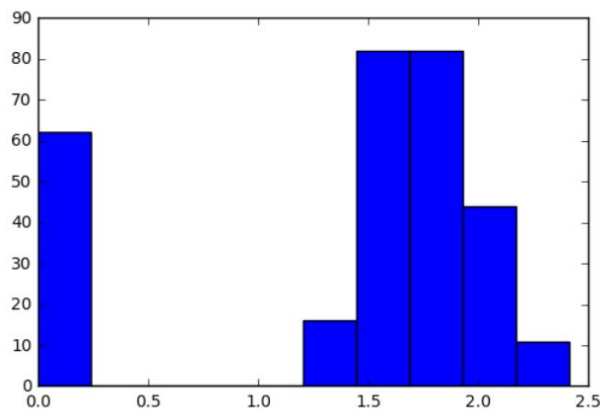
When you run it there will be a plot just like in 7. We could have made the H δ selection greater than -90 and the D4000 selection greater than 0 but this helps keep everything nice and uniform.

Find the Bounds

We could jump right into making the D4000 vs. H δ plot, but how would we know what to make the x and y bounds? Throwing in random numbers can work just fine, but that'll take too long. Instead, let's make histograms that automatically generates the ranges we need.

Make a variable that points to the quiescent selection of galaxies from D4000 that are greater than 0. Then, select out the quiescent galaxies with measured D4000. The last line of code builds the histogram, just as plt.scatter did in the last section.

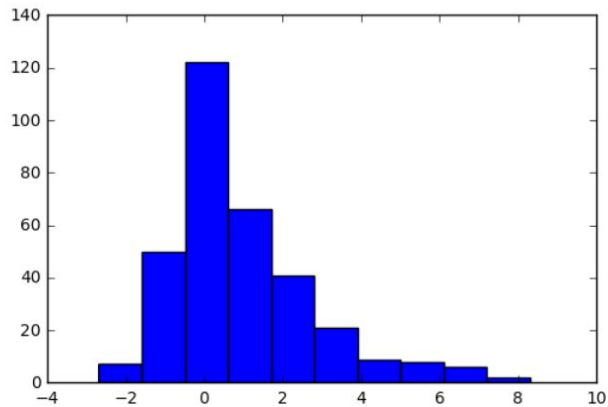
```
In [10]: d4000_selection = [data['D4000_N'][quiescent_selection] > -99]
d4000 = data['D4000_N'][quiescent_selection][d4000_selection]
n,bins,patches = plt.hist(d4000)
```



As you can see here, we've got points that extend to 0.0 and 0.25. However, these would best be considered outliers, so instead we want our bounds for D4000 to start at 1.2. We could go to 2.6, but 2.5 is sufficient.

Now make the histogram for H δ . This code is almost identical to that of D4000. The only difference is that you'll make variables that point to H δ and use values greater than -99.

```
In [11]: HDelta_selection = [data['LICK_HD_A'][quiescent_selection] > -99]
HDelta = data['LICK_HD_A'][quiescent_selection][HDelta_selection]
n,bins,patches = plt.hist(HDelta)
```



D4000 vs. H δ

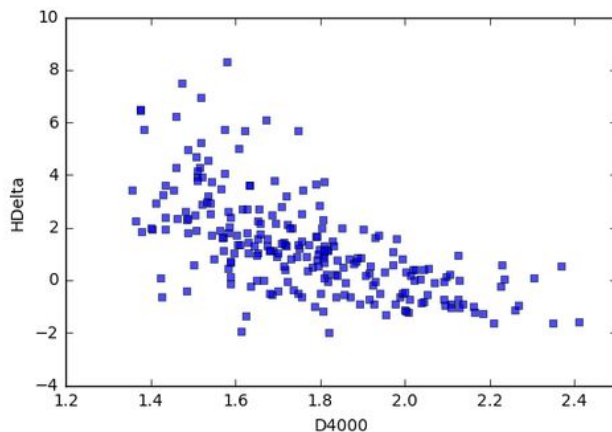
Using the variables we've already created, make two more that hold the H δ and D4000 data points that contain quiescent galaxies and values greater than -99.

```
In [12]: HDelta = data['LICK_HD_A'][quiescent_selection][HDelta_D4000_selection]
D4000 = data['D4000_N'][quiescent_selection][HDelta_D4000_selection]
```

Finally, build the scatter plot for D4000 vs. H δ . The x value will be D4000 and the y value will be H δ . Your x and y bounds are derived from the histograms we made earlier and the x and y labels are fairly self-explanatory.

```
In [13]: plt.scatter(D4000,HDelta,marker='s',s=16,alpha=0.7,linewidth=0.5)
plt.xlim(1.2,2.5)
plt.ylim(-4,10)
plt.xlabel('D4000')
plt.ylabel('HDelta')
```

```
Out[13]: <matplotlib.text.Text at 0x7f9af3cfb850>
```



Et voilà, our scatter plots are complete! Let's go plug them into Glue.

Generate Plots in Glue

Before we dive into Glue, you'll need to download a new [catalog](#). However, it's not really "new" per se. Rather it's the LEGA-C catalog but with quiescent galaxies only.

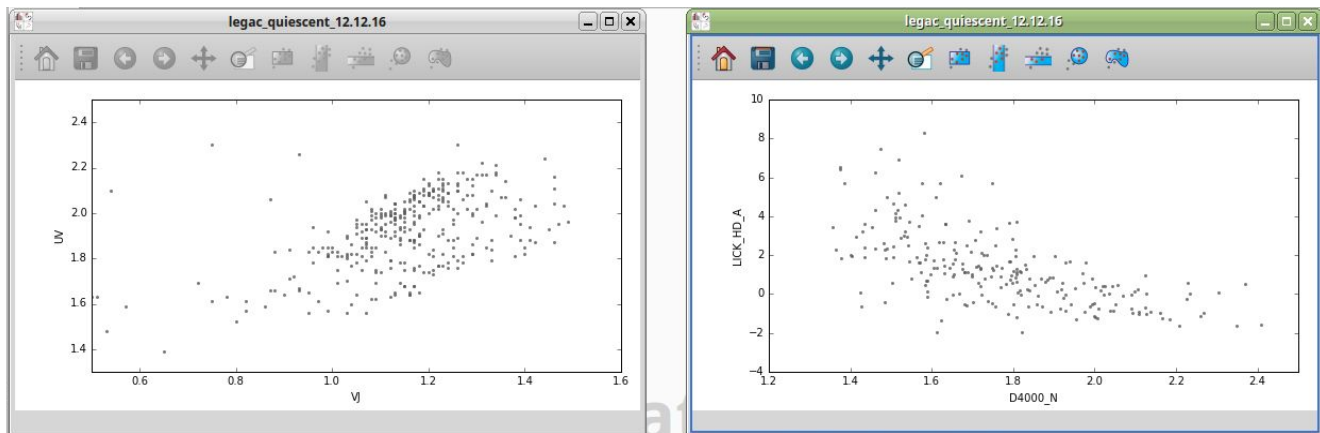
Return to your terminal and type `glue` into the command line. The Glue app will open up after a few seconds. A much more in-depth tutorial can be found [here](#), but here's a quick rundown. On the upper left is the *data manager* (or *data collection* if you're using a Mac), where open data sets and subsets will be listed. The section under "Plot Layers" is called the *visualization area* - this is where the tools to modify the appearance of your data reside. Then, there's the big white area where you see "Drag Data To Plot" in gray. This is called the *visualization dashboard*, and this is where you'll be able to see the plots, histograms, or images you make.

To open data, you can press `ctrl-o`, the red folder icon underneath the data manager, or the `File->Open Data Set` menu. Select the new LEGA-C catalog. Once you've done so, it'll show up highlighted in the data manager with a gray circle next to it. Click on it and drag it to the visualization dashboard. There's a few different data viewers you can play with, but we're just going to use "scatter plot" for now. As of this step, your screen should look something like this:

`gac`

That's a pretty nice scatter plot, but it's not terribly useful when it looks like that. Our original x and y axes are U-V and V-J, so go on over to the visualization area and change "id" to "vJ". We also need to fix the bounds so go back in the tutorial and see if you can find them (hint: the x min is 0.5). If done correctly, this will look identical to the plot we built in Jupyter.

The next plot is the D4000 vs. $H\delta$ plot. Just like before, drag the catalog to the visualization dashboard and select "scatter plot". Change the x and y axes to `D4000_N` and `LICK_HD_A` and switch the x and y bounds to the ones we got from our histograms. You should now have two scatter plots that look like this:



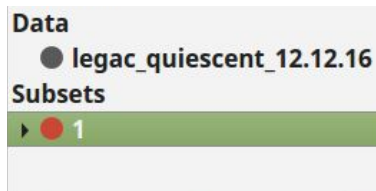
Create Subsets of Data


We can now define subsets of data on one plot and see where they show up on the other. The image window above the plots gives five options.



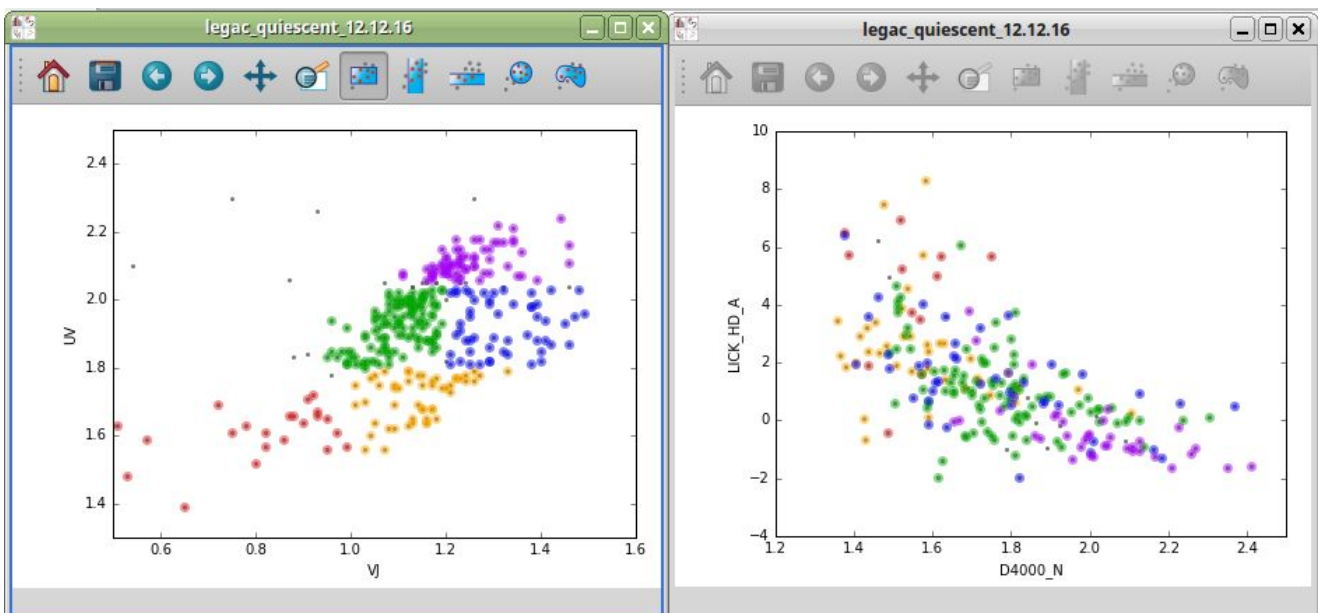
The first one lets you define rectangular sections. The second defines vertical sections all the way from the top of the plot to the bottom. The third defines horizontal sections all the way from the left to the right. The fourth defines circular sections. The fifth defines polygons of any shape you wish. Just make sure to press enter when you're finished defining your polygon, or escape if you wish to redraw it.

Let's use the rectangular section to make things easy. Go to the Toolbars menu at the top of the screen and make sure Replace Mode is selected. Now, click on the V-J vs. U-V plot and select an area that comprises some part of the bottom left corner. I started at the bottom left and went to about (V-J = 1.0, U-V = 1.7). You should now see points highlighted in some color (probably red) on both graphs as well a new subset. If you decide you want to change its location, make sure the subset is highlighted like this:

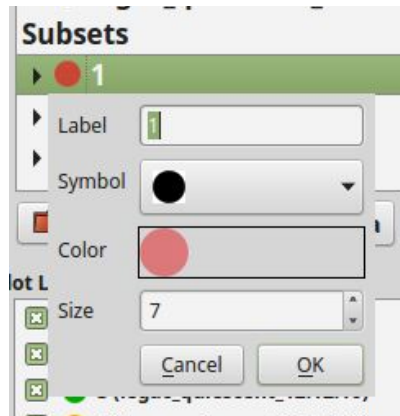


Otherwise you'll just keep making new subsets. You can also press ctrl and click the subset on the plot in the visualization area to drag it around. If you'd like to remove data points from the subset, you can go to Toolbars->Not Mode. Or, if you'd like to remove a subset altogether, press the  button.

Now that we've drawn up one subset, let's do a few more. Five in total should be sufficient to help us make some sort of correlation between the two plots. You can place them wherever you like, but it will probably be most helpful to make subsets that cause the plots to look like this:



If you wish to change the colors of the points, you can go up to the data manager and double click on the number of a subset to bring up a dropdown menu.



With this, you can change the label name, the symbols used for the subset, the color, and the size of the symbol.

Finally, if you wish to save your session, you can click File->Save Session. When you reopen Glue, you can go to File->Open Session to continue working again.

What can we learn from these plots?

For quite some time now, it's been known that a redder galaxy can sometimes appear red due to an abundance of dust. Our goal in plotting two age diagnostics, the strength of the 4000 Angstrom break (D4000) against the absorption line strengths ($H\delta$) of A-Type stars (which are typically one gyr), is to compare them with a color-color diagram and find out if age, rather than dust, is the cause for this redshift.

The Glue [website](#) states that “Glue uses the logical links that exist between different data sets to overlay visualizations of different data, and to propagate selections across data sets.” So in essence, Glue can take any points you highlight in one plot and link the data to another plot. With this, we can be sure that the colors do, in fact, correlate with the age diagnostics presented.

Once we start playing around with the data in Glue, we can clearly see that redder galaxies, which are in the upper right corner of the UVJ diagram, align with points showing older galaxies in the other plot. The D4000 vs. $H\delta$ plot shows an anticorrelation, so based on what we've seen, galaxies with stronger 4000 Angstrom breaks and weaker $H\delta$ absorption line strengths will appear older. Knowing that this data is obtained using *spectroscopy*, which is looking at the wavelength of light that each element in something absorbs or releases and separating these wavelengths into a spectrum to determine their composition, we can be certain that it is more accurate than other measures.