Homework 3: matplotlib/plotting

due Sept 25, 2013 @ 5pm

make a repo:

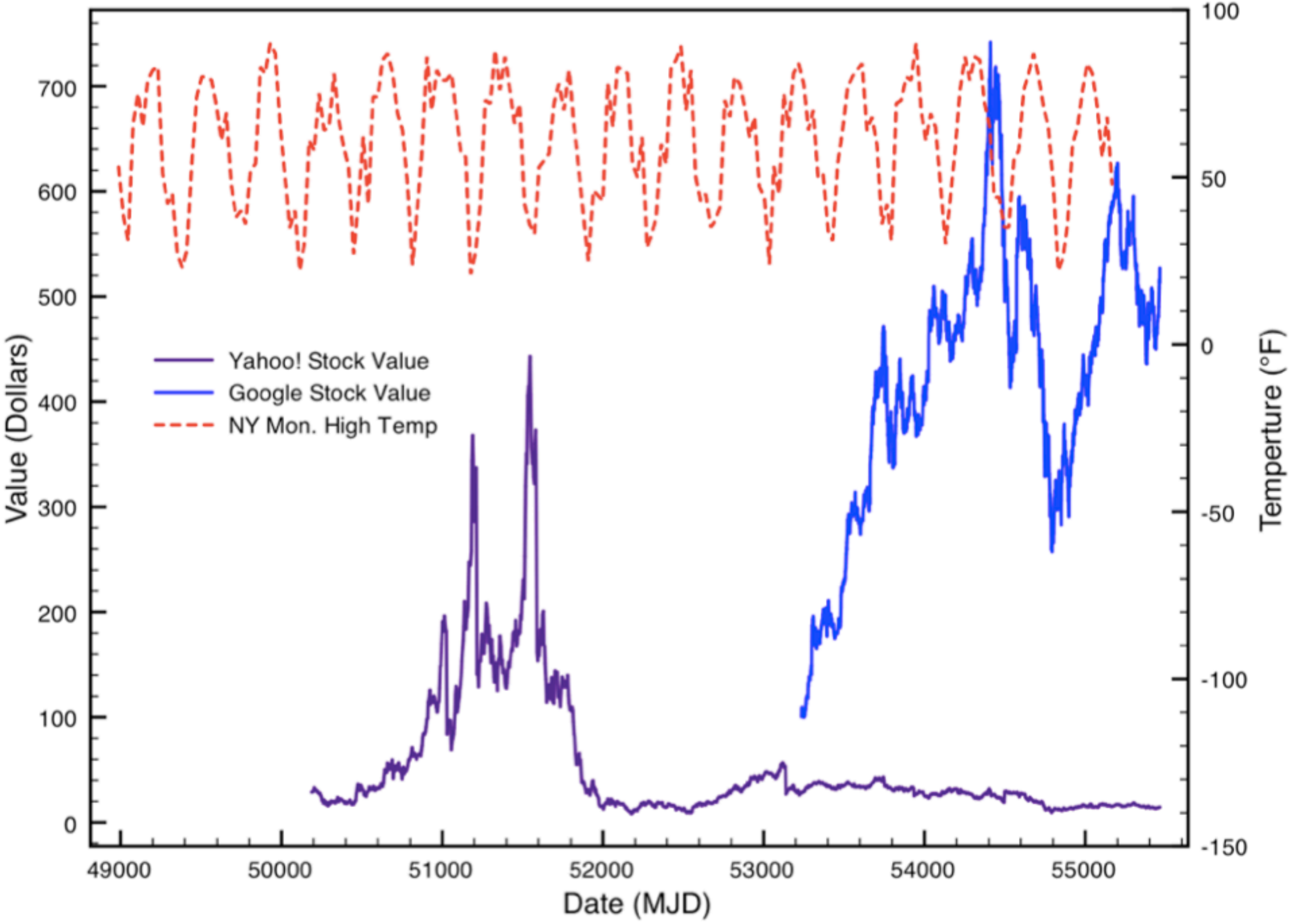
`github.io/<username>/<last_week_repo_name>/hw3`

and add us (amorgan, profjsb) if it's private

1) **Reproduce one of your old published-paper quality plots with `matplotlib`.** Provide the original plot, the recreated matplotlib plot, and the Python code used to make the new plot. You can ask us for an example from us if you dont have your own plots (we'll put a few on Piazza in a few days).

2) **Reproduce in `matplotlib` the provided plot `stocks.png`**
Use the provided datafiles `ny_temps.txt`, `yahoo_data.txt`, and `google_data.txt`. Provide your new plot and the Python code.

New York Temperature, Google, and Yahoo!



3) **Make a generic "Brushing" code**

It is obviously difficult to visualize data in plots when there are more than two dimensions or parameters to explore. Brushing is a technique that allows you to plot many pairs of parameters of a dataset in many subplots: when you highlight some region of data in one plot the associated data points in the other plots are automatically "brushed" (they change color, opacity, etc.). This allows you to highlight a region of interest (or maybe just an outlier) in some parameter space and see where that region lies in other dimensions.

Here, we'll ask you to make a brushing code that can read in a dataset of with many rows and multiple variables/parameters (columns), make subplot of some pairs of parameters, and then let the user interactively draw rectangular brush regions. After the rectangle is drawn, the data not associated with that region has it's opacity reduced in **all** subplots. All the user has to do to remove a brushed region is to type "d" while the mouse is over that brushed region.

Make a generic "Brushing" code

Some starting ideas:
1. First make sure you can make subplots in `matplotlib`. Try reading in a dataset as a record array using `loadtxt`

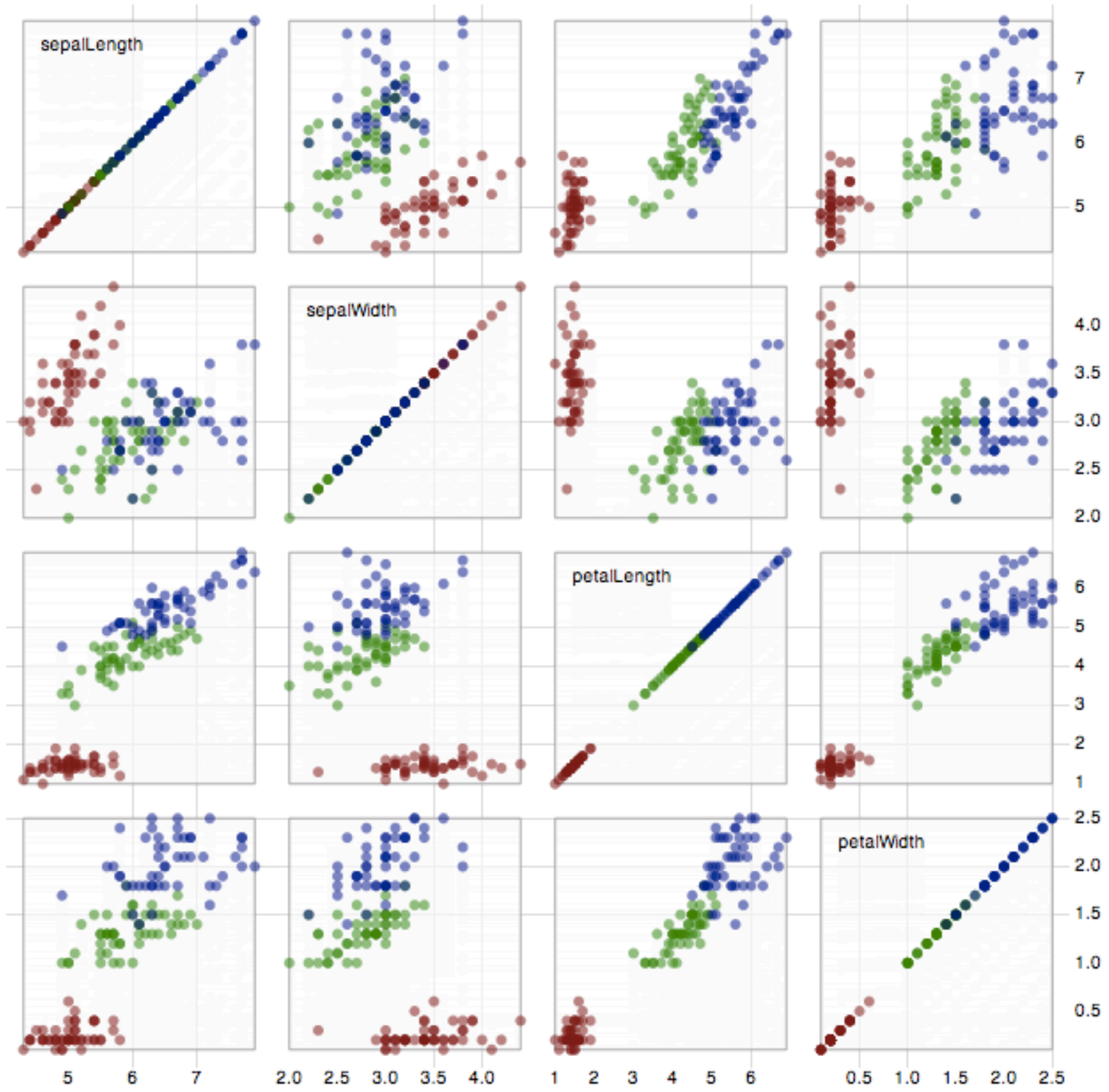
2. write some class-based code to draw and save rectangular regions
http://matplotlib.org/api/artist_api.html#matplotlib.patches.Rectangle
You'll also want to play around with registering events in `mpl`.

3.To get a better idea of what we're looking for, you might want to look at the project "`viewpoints`" or "`GGobi`"
http://www.assembla.com/wiki/show/viewpoints/

`http://www.ggobi.org/demos/brushing-simple.html`

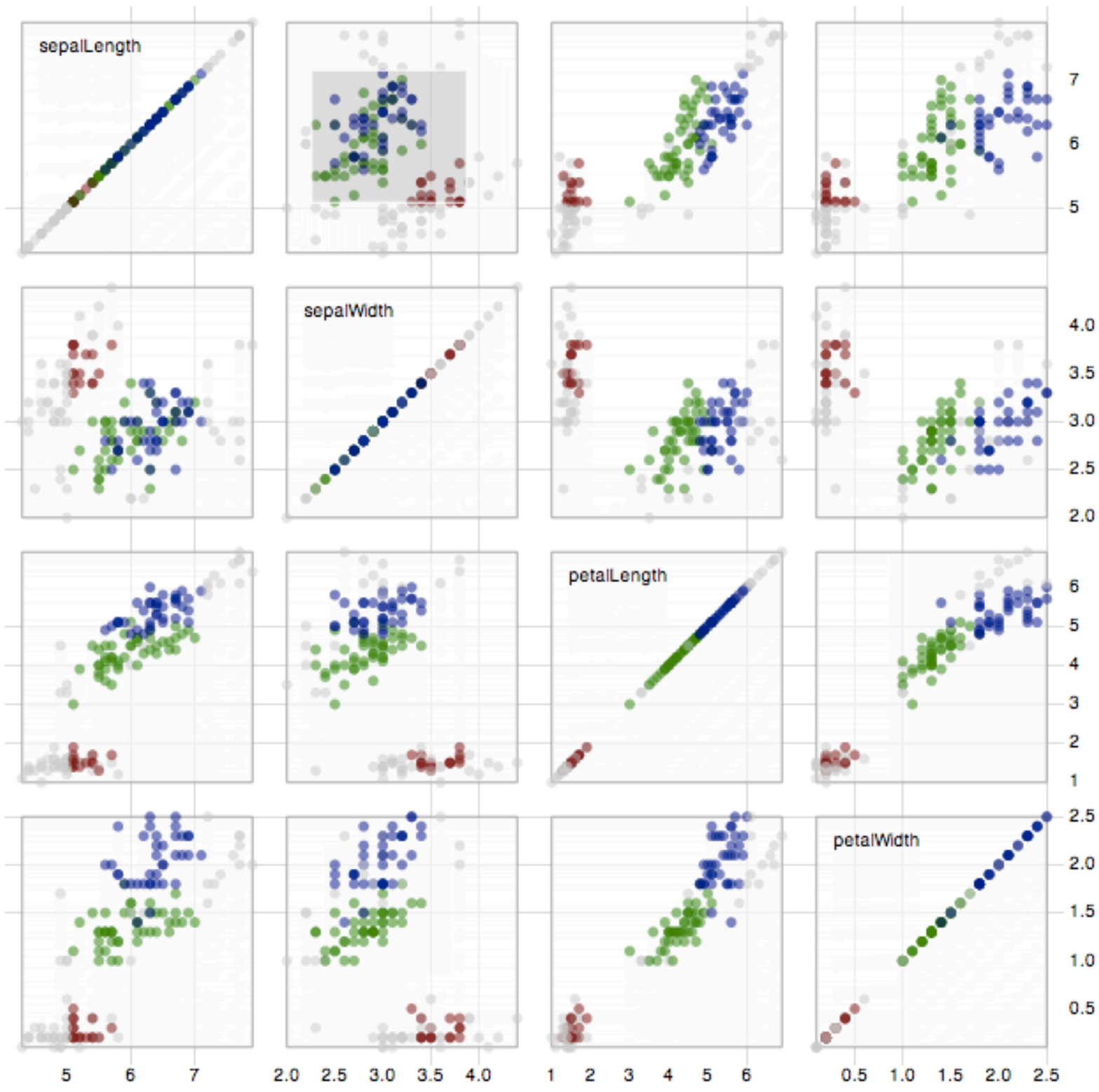
`http://mbostock.github.com/d3/ex/splom.html`

Brushing example:



http://mbostock.github.com/d3/ex/splom.html

Brushing example:



http://mbostock.github.com/d3/ex/splom.html