Computational Optimization
Homework 2

Nedialko B. Dimitrov

By completing this homework assignment you will learn some basics of:

- Pandas – Python data reading and processing package.
- Matplotlib.basemap – A Python visualization package for geographic data.

Pandas is an extremely popular package for reading and searching medium sized data sets, up to around 500k rows though it can work on larger ones as well. Matplotlib.basemap is a light-weight, low-overhead geographic plotting package. There are other geographic plotting packages, like the python interface for qgis, but they are more complex to use. To help make the geographic plotting easier, I am providing you with code I wrote for a geoplotter package that simply encapsulates some of the Matplotlib.basemap plotting functions.

Execute the following steps, and explore the package documentation as you go to learn more. By completing these steps, you will create a video similar to the one posted on the class webpage.

1. Use geoplotter to create a basic plot of the world. This should come up just by calling the right functions of geoplotter, and you shouldn't need to do much extra work.
2. Download the National Material Capabilities data set from here:
   http://www.correlatesofwar.org/data-sets/national-material-capabilities
   If you like, read the codebook to understand what the columns of the data set mean.
3. Write some python code to: 1) read in `NMC_v4_0.csv` into a Pandas data set 2) Given a year, and a COW Country Code, be able to pull out the CINC of the country 3) Get all the unique COW Country Codes.
4. Download the world borders shape file `cshapes_0.4-2.zip` from:
   http://downloads.weidmann.ws/cshapes/Shapefiles/.
   After unzipping, use the `readShapefile` method on geoplotter to read in the shape file. Look through the `_info` variable to see what pieces of data are associated with each shape. Specifically, the COWCODE data for a shape tells you what country that shape belongs to. Use `drawShapes` to draw the United States in a different color be sure to 1) write a loop to collect all the shapes associated with the US 2) draw all of those shapes in the color you chose.

You are going to create a movie visualizing the CINCs in the dataset over time. At a basic level, we are just repeatedly going to find a CINC, and plot a country in a different color. However, we have to do that in a few encapsulated steps.

1. The data set does not have every country in it every year. Create a class derived from the GeoPlotter class called MilexPlotter – the new class should automatically read in the data from the NMC database and the world country shape file. Over-ride the `drawWorld` method so that it draws all the continents in blue by default. That way, countries that are not in the data set will show up in blue color by default.

2. To MilexPlotter, add a method called `def plotCountry(self, code, **kwarg)` that takes in a COW Country Code, and plots that country given some keyword arguments. You are going to pass all the keyword arguments you get to the `drawShapes` method of geoplotter. That way, without doing any extra work, you'll be able to select what color your country should be in.

3. to MilexPlotter, add a method called `def plotYear(self, year)` that takes in a year, and plots all the countries that have a CINC for that year, in an appropriate color. To do this 1) clear the geoplotter 2) draw the world 3) iterate over the countries and plot them in a color corresponding to their CINC. The hardest part of this is figuring out what color corresponds to the CINC. We are going to do this by learning how to map numbers to colors in matplotlib. Let `maxCINC` be the maximum CINC for any country in the whole data set. Use `matplotlib.colors.Normalize` to map any number between 0 and `maxCINC` to a number between 0 and 1. Then, use `matplotlib.cm.hot` to map that number into a color. The normalization ensures that as you go through the years, the colors you plot are consistent with each other. Finally, your `plotYear` method is going to add some annotation to the plot to tell us what year it is.

4. Write a loop that repeatedly calls `plotYear` to plot all the years, and save each into a separate `png` file.

5. Now that you have all the frames of the video, lets put them together. Create a text file called `frameslist.txt` and then run the following command

   ```
   mencoder "mf://@frameslist.txt" -mf fps=3 -o cinc.avi -ovc xvid
                                   -xvidencopts fixed_quant=2:autoaspect
   ```

   The command simply takes all the frames and constructs a video called cinc.avi out of them. If you prefer, you can do this with whatever video program you know.

What to turn in: For this assignment, turn in two things:

1. Your Python code, in a file called `hw2.py`.
2. A single AVI file with the video you generated, in a file called `cinc.avi`.
3. A short (less than one minute) explanation of your code. Create this in the same way you careted your previous explanations and name it `hw2-explanation.avi`.

Key take-aways: From this assignment, you should take-away these things:

1. A bit about Pandas, and how to navigate its documentation to find the things you need. Mostly, you should now feel comfortable selecting rows out of a data set you have. Pandas is super useful also in combining data across different sets etc. You can read more about it when you need it online.

2. A bit about the way that geographic data is structured. You now know shape files, and how to work with them. You also know one way of plotting them.

3. More complex encapsulation. In this assignment, you created your own object, based on an existing object given to you. In doing so, you took advantage of code someone else wrote, and adapted it to your needs. The key part of this is that you didn't cut-paste, so now if someone adds some great functionality to geoplotter, your object has it automatically as well.