ClimateVis: an interactive web app

Hosted by Github at https://github.com/ayocom/climate-vis.

Gene Krasnitskiy, Rudy Resch, Simon Wiedemann, and Andrea Yocom

June 13, 2014

Contents

1 Summary of ClimateVis Web App

Our team has developed an interactive visualization of average annual temperature, total annual precipitation, and total annual carbon dioxide emissions. There are three tabs: Map, Correlations, and Rates.

1.1 Map Tab

In the Map tab, users can watch the evolution of one or more of these variables at a time on a Google map. Users can also draw a rectangle around a region to produce a line plot of each variable within the region over time.

1.2 Correlations Tab

In the Correlations tab, users can manipulate both axes of a 2D scatterplot in order to explore correlations between variables. The variables that can be chosen are year, average annual global temperature, total annual global precipitation, total annual global anthropogenic carbon dioxide emissions, and average annual Arctic sea ice extent.

1.3 Rates Tab

In the Rates tab, users can search for correlations in the slope of each variable. These slopes are computed by finding a linear fit to 6 points at a time in a similar method as boxcar smoothing.

1.4 See It in Action

To see our web app, clone our Github repository. In order to run the project, you will need to have installed Flask:

```
pip install flask
```

Pip is a package manager for python. If you do not currently have it, it needs to be installed as well. Now navigate to the climate-vis/website directory and run

```
python climate-vis.py
```

After starting `climate-vis.py` you can navigate to `localhost:5000` in your web browser to view the project.

2 Project Design

2.1 Original Project Goal

We originally aimed to develop an interactive visualization that allows users to explore what correlations exist between atmospheric CO2 concentrations and hydrological variables related to global climate change. We intended to use data from 1850-present of the following hydrological variables: sea level, ice volume, and oceanic temperature. The original concept was to visualize correlations in a moving bubble chart, in which time is represented by moving frames that will trace out a pattern (see for example Gapminder).

We had several additional interests. These were the following:

1. To build an intensity worldmap visualization that allows users to explore spatiotemporal patterns in precipitation levels worldwide.
2. To make projections based on historical data for hydrological variables.
3. To explore non-hydrological variables, particularly economic datasets such as GDP and energy consumption, which could be plotted both on the intensity worldmap and on the moving bubble chart.

2.2 Project Clarification

As we hunted for data for this project, the desire to build an intensity worldmap engulfed our original goals. We were able to find temperature, precipitation, and carbon dioxide emissions data which were mapped onto a latitude/longitude grid. For this reason, we focused first on the intensity heatmap and correlation plot for these variables, rather than sea level or oceanic temperature variables.

3 Data

3.1 Data Sources

We found total monthly precipitation in millimeters[1] and monthly air temperatures in degrees Celsius[2] that had been projected onto a 0.5 degree latitude and longitude grid. These data are from measurements at stations, and are carefully verified and cleaned prior to release.

We also found a one degree latitude by one degree longitude grid of carbon dioxide emissions from anthropogenic sources in units of million metric tons of carbon per year[3]. These data are based on historical records of energy production, energy consumption, and trade, and are mapped onto a latitude and longitude grid by averaging over political units represented by 1984 population distributions.

3.2 Data Manipulation

The manipulation of the precipitation and air temperature data was quite simple. The files were well-organized, though large. Python dictionaries and Pandas dataframe objects were both used for the organization of this data. The manipulation of the carbon dioxide was slightly more involved; Gene wrote a script to map the 1D list of data onto an actual grid.

3.3 Data Reduction

We quickly realized that the precipitation and air temperature data was a massive set. Each was around 1GB. Since Github limits us to 2GB total storage, we needed to reduce the dataset.

One such data reduction was to convert the precipitation and temperature data from 0.5 degree to 1 degree latitude and longitude resolution. Since the carbon emissions data were provided on a 1 degree grid, this reduction step made sense.

A second data reduction measure was to compute the annual average of air temperature data and the annual total of precipitation data at each latitude and longitude point. This reduction produced 40-50 MB files that were reasonable to deal with in the app. On

[1] Willmott, C. J. and K. Matsuura (2012) Terrestrial Precipitation: 1900-2010 Gridded Monthly Time Series (1900 - 2010), http://climate.geog.udel.edu/~climate/html_pages/Global2011/Precip_revised_3.02/README.GlobalTsP2011.html.

[2] Willmott, C. J. and K. Matsuura (2012) Terrestrial Air Temperature: 1900-2010 Gridded Monthly Time Series (1900 - 2010), http://climate.geog.udel.edu/~climate/html_pages/Global2011/README.GlobalTsT2011.html.

[3] Carbon Dioxide Information Analysis Center (CDIAC), **DOI:** 10.3334/CDIAC/ffe.AnnualIsomass.2013

the other hand, we lost interesting information about monthly fluctuations that previously existed in the temperature and precipitation data. If the project were to continue, we would look into hosting data elsewhere (potentially on Dropbox instead of Github), to avoid this limitation. This problem actually proved to be a major sticking point for this project.

3.4 Data Features and Conclusions

One of the questions that prompted our query was: does global warming lead to increased precipitation in wet places and decreased precipitation in dry places? To investigate, we could filter data to look at places where annual total precipitation is above a certain high threshold, or those for which it is below a certain low threshold. The former could be considered wet places and the latter dry places. It was interesting to discover that the trend for these is pretty much the same - precipitation decreased over the past century for wet as well as for dry places. It is difficult to conclude from the data so far whether this is a cyclical trend and precipitation will increase again, or if there is a new normal for the annual precipitation. As expected, average annual temperature has risen over the century, as we also observe decrease in the extent of Arctic sea ice and exponential rise in global CO2 emissions.

It would be interesting to add other variables such as El Nino cycles, global sea level rise, ocean current distribution, etc. It would also be interesting to consider certain periods of time such as summer versus winter and enable the user to explore the trends in the variables observed over those periods in the Correlations tab.

4 Visualization Engine

We decided to visualise the data with heat maps. The heat maps show the spatial distribution of the data values at a specific year. We also decided to provide to the user with some control parameters and selection tools, in order to facilitate the qualitative analysis of the data.

4.1 Control parameters

At first, users are able to select/remove the desired heat map. They may also represent more than one heat map at the same time. Second, users may select the data for a year in between 1900-2010. They may also animate the data by selecting the years passed per step and the time it takes to update the heat map. Finally, users may zoom into the map, change the opacity of each heat map and change the radius of influence of each data point.

4.2 Rectangle tool

The user is able to select with the rectangle tool a region on the map. The average data values inside the rectangle will be calculated and a line chart displaying the average values over the years will show up under the map. Also, the calculated averages will be displayed in the text box that is located at the bottom of the page, for easy selection in the case that analysis external to the web app is desired.

4.3 Information at a specific point on the map:

Right click on a specific location on the map, and the values of the data will show up in an info window on the map.

4.4 Improvements

We would like to suggest some improvements to the visualization engine. The first problem we noticed is that the change in the average air temperature over the years is almost not noticeable, due to the method by which the Google API library represents the heat map colors. The Google library assigns to each data value a color from the selected gradient, depending on the weight magnitude and the density of data points. Google takes the minimum and maximum value of the data, and interpolates its range with the colours of the gradient. Since the temperature data has a wide range (approx. -60 to 40 degrees Celsius), and since the average temperature fluctuations are comparably small (before the year 2000, approx. 2 degrees Celsius and after, 5 degrees Celsius), the color in the region does not noticeable change.

Second, it may be possible to improve upon the animation speed performance. Third, it is necessary to incorporate a color legend (the Google maps API still doesn't provide that feature for heat maps). Finally, the user should be able to draw more than one rectangle on the map and display the information of several regional values at the same time, in order to enable regional comparisons.

5 Web App Framework

The web app framework for this project was designed using Bootstrap and Flask. Bootstrap takes care of a majority of the layout involved with producing webpages; specifically, it supplies an easy to use way to create gridded content organized by columns and rows. The template (dashboard) for laying out the menus and content was supplied as a basic example for Bootstrap, which then needed to be extended for our project. Another feature that Bootstrap provided was the ability to selectively mask components depending on the size of the browser window. This is helpful when you need to view the app on a smaller screen, or need to snap a window to fill only half of a screen while retaining a clear view of page content.

Flask is a web framework for Python, and it was chosen as a simple system to reduce the amount of code copying for elements that would be used on multiple pages. This is accomplished by storing all of the navigation elements and global Javascript in a base template, and separating the remainder of the content into simpler pages. Consequently, changes in the global framework would not need to be patched to each individual page. In addition, Flask allows future extensions of the project to use actual Python code in order to perform data manipulation and analysis such as fitting and future projections. This would also allow us to replace some of the Javascript code with more flexible Python scripts. The Flask framework also allows simplification of the URLs for a web app by creating paths to the actual HTML code displayed on the page. Given time to implement some of the improvements/extensions to the project, this would allow us to better organize the app as the number of pages increased.