Process Book

Air China 2.5

Visualization of Chinese Air Pollutant PM2.5

https://syuanivy.github.io/AirGlobal/

Revised Project Proposal

Basic Info

- Project title: Air China 2.5
- Team members:
  - Wenhan Rao, wrao2@usfca.edu, 20337018
  - Xitao Wang, xwang109@usfca.edu, 20318577
  - Shuai Yuan, syuan6@usfca.edu, 20336937
- Github repository：https://github.com/syuanivy/AirGlobal.git (branch: final)
- Project URL: https://syuanivy.github.io/AirGlobal/

Background and Motivation

Despite increasing efforts to combat it worldwide, air pollution has been a growing problem threatening human health, the ecosystem and global economy. Especially for developing countries, air pollution is believed to cause more deaths than many lethal diseases and viruses such as AIDS and malaria. In China, air pollution was estimated to cause 1.2 to 2 million deaths annually, primarily by exacerbating cardiorespiratory diseases [1]. There is a broad spectrum of common air pollutants, including toxic gas such as CO, NO_2, Ozone, SO_2, lead and particulate matter. Smaller particulate matter with a mean aerodynamic diameter of 2.5 μm (PM2.5) is especially detrimental to health with a 36% increase in lung cancer per 10 $\mu g/m^3$ as it can penetrate deeper into the lungs [2]. People with breathing and heart problems, children and the elderly may be particularly sensitive to PM2.5.

PM2.5 has drawn international attention and more and more monitoring stations have been established to collect data on PM2.5 concentration, which is used for broadcasting and scientific studies. The US Environmental Protection Agency (EPA) and the US embassies/consulates across the world have been monitoring air pollution and make the data available in various formats. As a fast developing country, China has suffered shocking increase in PM2.5 in the past few years and the government started a national Air Reporting System that now includes 945 sites in 190 cities. Hourly air pollution data including PM2.5 concentration are now available to the public. It shows the the most polluted area is the east of the country and it is widespread across central and northern China.

Project Objectives

This project aims to provide an interactive web service that allows users to explore global air pollutants data. Datasets from different sources were obtained for various purposes and thus are good for illustrating different discoveries. This project expects to integrate data from different sources and let the user to choose datasets as well as the visualizations according to their individual interests such as region, time frame, severity and etc.
JavaScript and D3 will be used for implementation. Other libraries such as TopoJSON, toGeoJSON and etc. are also likely to be utilized.

Data

At this stage, we are considering integrating as much available data as possible, however the feasibility with certain extremely big data from the satellites needs to be evaluated during implementation. Figure 1 shows the four major data sources we consider to use.

Figure 1. Data sources used in the project.

AirNow is a US government web service showing Air Quality Index (AQI) for all the states (https://airnow.gov/index.cfm?action=google_earth.index). It is available in the format of Keyhole Markup Language (KML) files and can be viewed in 3D viewers like Google Earth. To use it in this project, we will probably need to transform it into format such as GeoJSON.

EPA (US Environmental Protection Agency) provides data for a variety of air pollutants including CO, NO_2, Ozone, SO_2, lead, PM2.5 and PM10 (https://www3.epa.gov/airdata/ad_maps.html). The data covers area where US embassies/consulates are located across the world. Thus gives a good overall representation of global air pollution. The data are available in in KMZ, which are zipped KML files, and thus may also need to be transformed.

US_mission_China provides hourly PM2.5 data in five big cities of China over a period of 5 to 8 years (http://www.stateair.net/web/post/1/5.html). It is a great resource to show how China major cities have been suffering increasing air pollution and records dramatic peaks of the worst days in history. The data is in plain text and do not need to be transformed.

Berkeley Earth collected the most detailed information about China air pollution both temporarily and spatially(http://berkeleyearth.lbl.gov/manual/china_air_quality/). Data was collected and preprocessed into gridded time-series data in eastern China, which has suffered the worst PM2.5 pollution in the world. However, the data is extremely large and in Network Common Data Form (NetCDF or .nc file), which will need to be transformed as well.

Data Processing

First, data in KML or NetCDF needs to be transformed into formats readable by our JavaScript libraries.
Second, the data quality needs to be verified because there are lots of monitoring stations giving problematic data due to hardware malfunction. We need to eliminate the bad data and use only the verified data.
At last, if we decide on implementing color gradient data with contour lines, we will need gridded data (concentration value for each latitude-longitude pair across a certain area). In that case, we will need to generate interpolated data using the metadata (latitude-longitude) of the monitoring stations. That can be done with known interpolation algorithms [1].

Visualization Design

Prototype 1: Discrete labeling on 2D map

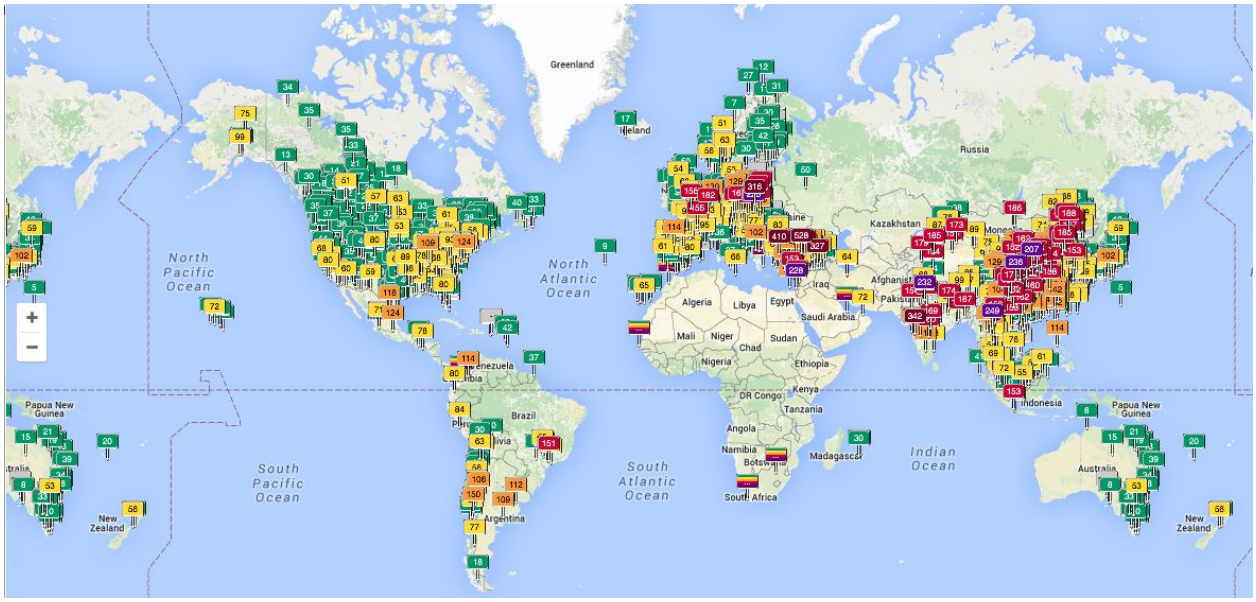


Figure 2. Discrete data point are labeled on a 2D map [3].

On a 2D world map shown in Figure 2 [3], labels all the monitoring stations according to metadata (latitude-longitude pair) with average PM2.5 concentration. Users can zoom in and zoom out the map to select area of interest. Upon clicking each label, the metadata about each monitoring station will be shown. Further clicking the label will show time-series data, if available, obtained at that station.

This kind of visualization requires the least amount of data processing as no interpolation is required. However, it is not so pretty as the data points are discrete and not evenly distributed over any area. In certain areas such as big cities and developed countries, the monitoring stations are very dense, while in the vast rural area or countries that have not yet started monitoring the data points are very sparse.

Prototype 2: Color gradient heat map

To avoid having the problem mentioned in prototype 1, we can preprocess the data by generating interpolated gridded data. However, we need to carefully select the algorithm used to estimate the concentration in area where pollutants concentration are not directly measured. The idea comes from [4] and is shown in Figure 3.

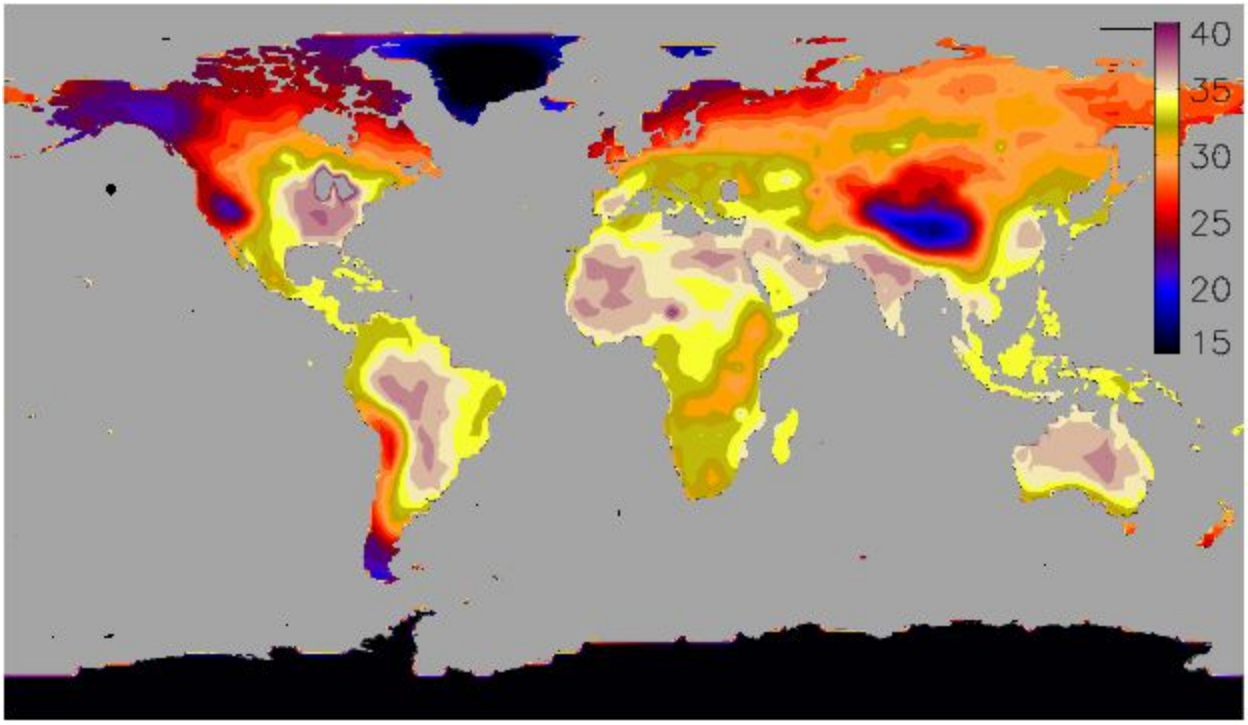


Figure 3. Color gradient heat map using interpolated data [4].

After generating gridded data at high enough resolution, we can use color gradient heat map to show the average PM2.5 concentration. Data for a specific area will be shown upon mouse over. It should also support zoom in and zoom out, clicking on specific area should show time-series data of the area if available.

The problem with this is the preprocessing can be tricky and also the data will be extremely big for a simple web server. The feasibility needs to be addressed.

Prototype 3: 3D viewer

Since we have data in KML format, another idea is to use a 3D viewer such as the NASA Earth Now Android App [5]. Interpolated data will be necessary as discrete labeling will look very bad on a sphere. In this case, user interaction should involve rotating the earth in any direction and zooming according to user's area of interest. Clicking on specific region should show the data of the region and further clicking should show time-series if the data is available.

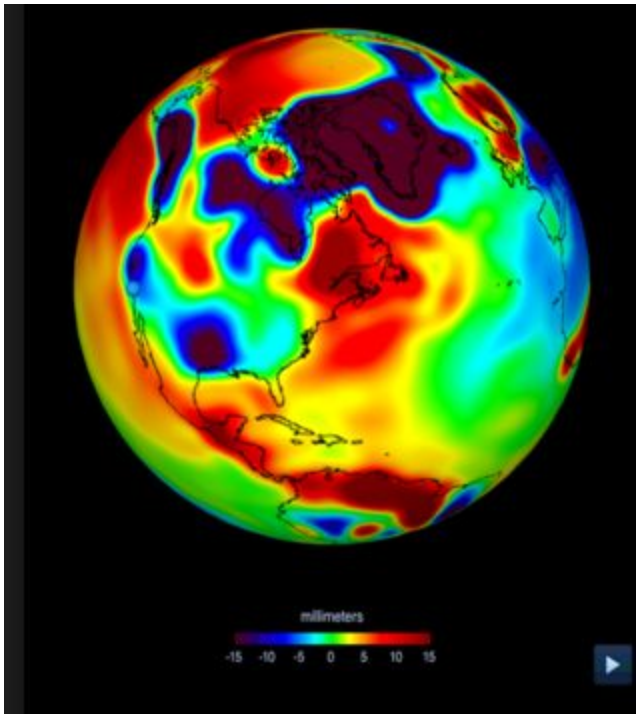


Figure 4. 3D viewer of global data [5]

Again the problem with this is the preprocessing and the size of the generated gridded data. The feasibility and necessity of such an approach needs to be addressed.

Current decision :

Time-series visualization

Besides basic visualization of average concentration on a map, we will provide interactive visualization of time-series data of specific city/monitoring station using line graph, and upon selection on specific time points, heat map of the selected time points of the same area should be shown, as shown in Figure 5.

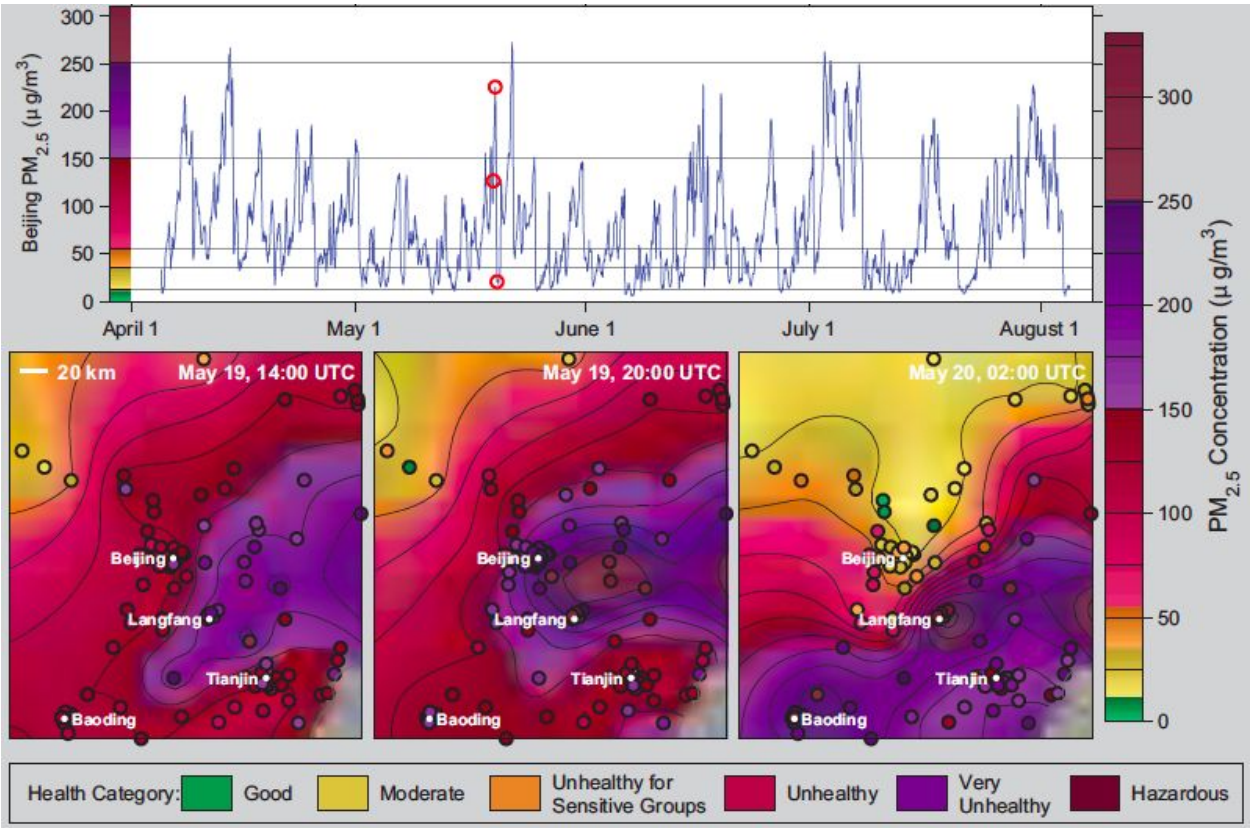


Figure 5 Time-series data visualization [1]

Bubble chart showing correlation with economy

Use bubble chart to show correlation of pollution with economy in major cities. Use area to indicate the economy and use color to indicate the severity of pollution, as shown in Figure 6 (http://www.improving-visualisation.org/visuals/tag=News) [6].

Search bar

Allow user to search data for a specific city or area by using a search bar to enter the location and show its pollution information.

Small multiples comparing spatial and temporal differences

Use small multiples to see major cities in the world and their pollution over a week/month or comparing the pollution in multiple cities shown in Figure 7 (http://jonathansoma.com/tutorials//d3/small-multiples/) [7].

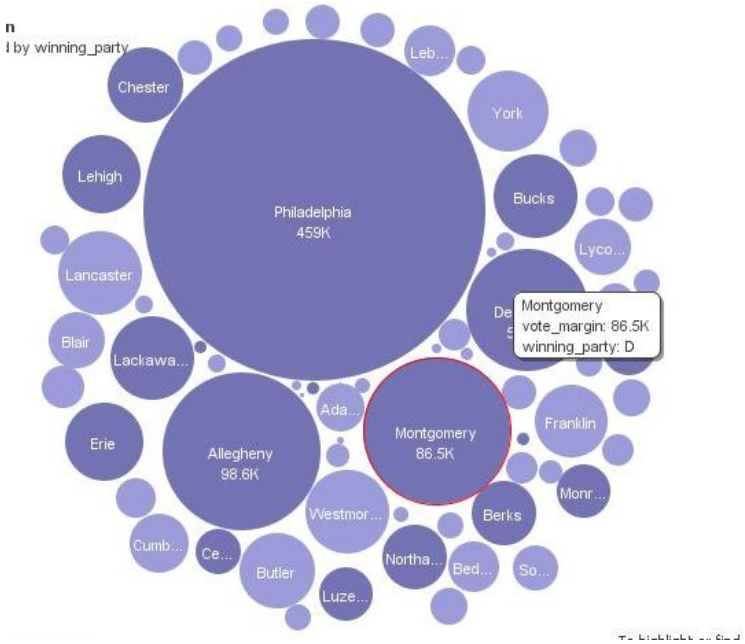


Figure 6 .Bubble chart showing correlation between pollution and economy [6]

Figure 7 .Small multiples comparing spatial and temporal pollution severity [7]

Must-Have Features

- PM2.5 concentration on 2D map (not as intuitive as Air Quality Index)
- Air Quality Index on 2D map
- Time-series data in line graph with zooming in and out
- Time points comparison upon selection
- Mouse over or on click interaction on specific location and time point
- Filter with values of concentration, air quality index
- Selection of area, time frame

- Ranking according to pollution severity/ AQI
- Search bar allowing user to search location-specific data
- Small multiples visualization comparing different locations and time points

Optional Features

- Dynamic display of time-series data upon selection of time frame (dropped after beta according to instructor's advise)
- Correlation of pollution severity among different area/countries (dropped after exploration, no meaningful correlation discovered)
- Bubble chart showing correlation of pollution with economy (dropped due to difficulties in getting economy data at such a high spatial resolution)
- 3D viewer (dropped due to data conversion)

Project Schedule

At the stage of design, the member responsibilities are not clear yet. It will be added later.

Week	**Plan**
3/21--3/27	Data collection, verification, project proposal
3/28--4/03	Data processing, updated proposal
4/04--4/10	PM2.5 average visualization, China on 2D map
4/11--4/17	α release, map visualization, dynamic display over time, search bar
4/18--4/24	PM2.5 Time-series for China regions and major cities, select area, zoom
4/25--5/01	β release, filter with concentration/time,, 3D viewer (dropped), small multiples
5/02--5/08	Integration, debugging
5/09--5/10(5/12)	Report, presentation, code clean up

References

[1] Robert A. Rohde , Richard A. Muller Air Pollution in China: Mapping of Concentrations and Sources. *PlosOne* 2015.
[2] Introduction to particulates pollutants https://en.wikipedia.org/wiki/Particulates
[3] Real time air pollution index http://aqicn.org/map/world/
[4] http://web.science.unsw.edu.au/~stevensherwood/wetbulb.html
[5] Earth Now: https://goo.gl/BqQcHT

[6] Bubble chart http://www.improving-visualisation.org/visuals/tag=News
[7] Small multiples http://jonathansoma.com/tutorials//d3/small-multiples/
[8] Huan Li, Hong Fan and Feiyue Mao A Visualization Approach to Air Pollution Data Exploration---- A Case Study of Air Quality Index(PM2.5) in Beijing, China. *Atmosphere 2016*
[9] NetCDF is a set of software libraries and self-describing, machine-independent data formats that support the creation, access, and sharing of array-oriented scientific data. http://www.unidata.ucar.edu/software/netcdf/

Data Cleanup and Processing

Berkeley Earth Dataset

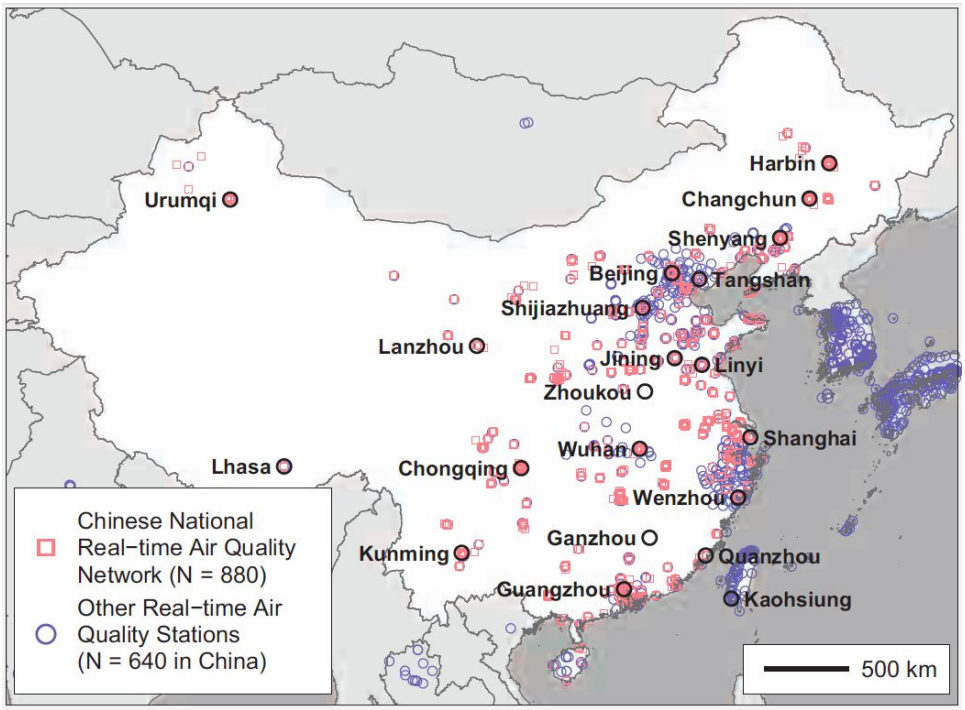
More than 600 monitoring stations covering many Chinese cities in the east part of the country, as well as Japan, South Korea and other pacific islands.

Problems:

- Data for certain cities on our map are missing completely, some are missing for certain time periods.
- Some cities have multiple monitoring stations thus have multiple PM2.5 values for a specific time point (See figure below).
- Some city names do not match the ones for the map.
- Some city names are not included on the map.
- City names associated with the map are in Chinese and cannot be indexed by the data model directly.

Things we did to solve the problems:

- Convert from .nc files to .csv files with a python script(not pasted here).
- Delete city data points not included on the map
- Translate Chinese city names into English to match the data.
- Combine multiple stations within the same city into one data point using the average.
- Mapping cities in the data to cities in the map with city IDs.



Details for Cleanup Berkeley Data

More than 600 monitoring stations covering many Chinese cities in the east part of the country, as well as Japan, South Korea and other pacific islands.

Problems:

This is just part of the original stations dataset, there are around 1800 stations. As you can see, for Beijing, there are about 30 stations. The work I need to do is that combine all stations city by city then combine this dataset with next dataset and get rid of all useless data based on our map for example, I delete Japan, South Korea stations from my dataset.

9	40.292000	116.220000	Changping Dingling, Beijing
10	40.217000	116.230000	Changping Town, Beijing
11	39.982000	116.397000	Chaoyang Olympic Sports Center, Beijing
12	39.937000	116.461000	Chaoyang Agricultural Exhibition Hall, Beijing
13	39.718000	116.404000	Huangcunzhen, Daxing, Beijing
14	39.929000	116.417000	Dongcheng Dongsi, Beijing
15	39.886000	116.407000	Temple of Heaven, Dongcheng, Beijing
16	39.939000	116.483000	East Fourth Ring Road, Beijing
17	39.742000	116.136000	Fangshan, Liangxiang, Beijing
18	39.824000	116.146000	Fengtai Yungang, Beijing
19	40.090000	116.174000	The Haidian northern New Area, Beijing
20	40.002000	116.207000	Haidian Beijing Botanical Garden, Beijing
21	39.987000	116.287000	Haidian Wanliu, Beijing
22	40.328000	116.628000	Huairou town, Beijing
23	40.499000	116.911000	Miyun Reservoir, Beijing
24	40.100000	117.120000	Donggaocun Zhen, Pinggu, Beijing
25	39.712000	116.783000	Yongledianzhen, Tongzhou, Beijing
26	39.520000	116.300000	Yufazhen, Daxing, Beijing
27	40.365000	115.988000	Badaling Northwest, Beijing
28	39.580000	116.000000	Liulihezhen, Fangshan, Beijing
29	39.937000	116.106000	Mentougou, Longquan Town, Beijing
30	40.370000	116.832000	The town of Miyun, Beijing
31	39.856000	116.368000	South Ring Road, Beijing
32	40.143000	117.100000	Pinggu town, Beijing
33	39.899000	116.395000	Qianmen E St, Dongcheng, Beijing
34	39.914000	116.184000	Shijingshan city, Beijing
35	40.127000	116.655000	Shunyi New Town, Beijing
36	39.886000	116.663000	Tongzhou New Town, Beijing
37	39.954592	116.468117	Beijing US Embassy, Beijing
38	39.929000	116.339000	West Park officials, Beijing
39	39.878000	116.352000	West Wanshou Nishinomiya, Beijing
40	39.954000	116.349000	Xizhimen N St, Beijing
41	40.453000	115.972000	Yanqing town, Beijing
42	39.795000	116.506000	BDA, Beijing
43	39.876000	116.394000	Yongdingmen Inner St, Beijing

This is a hourly dataset recording more than 1700 stations PM2.5 and more than 2800 hourly records. The work I need to do is that combining all stations for each city by computing the hourly average for that city.

	A	B	C	D	E	F	G	H
1	% Year	Month	Day	Hour	1			
2	2014	4	5	0	154.54	NaN	NaN	NaN
3	2014	4	5	1	144.59	7.2	33.02	54.99
4	2014	4	5	2	142.65	6	NaN	48.09
5	2014	4	5	3	142.65	5.04	37.94	18.76
6	2014	4	5	4	115.54	3.12	NaN	15.9
7	2014	4	5	5	115.54	3.12	39.97	11.04
8	2014	4	5	6	109.73	1.2	39.97	9.12
9	2014	4	5	7	74.87	1.2	33.97	8.16
10	2014	4	5	8	67.12	NaN	32.07	8.16
11	2014	4	5	9	88.42	1.2	33.02	9.12
12	2014	4	5	10	86.49	6	39.97	9.12
13	2014	4	5	11	101.98	6	42	9.12
14	2014	4	5	12	94.23	5.04	39.97	12
15	2014	4	5	13	74.87	7.2	NaN	NaN
16	2014	4	5	14	78.74	4.08	40.78	35.91
17	2014	4	5	15	92.3	4.08	40.78	42.81
18	2014	4	5	16	94.23	NaN	40.78	40.78
19	2014	4	5	17	96.17	NaN	31.12	74.87
20	2014	4	5	18	94.23	5.04	30.17	61.31
21	2014	4	5	19	90.36	4.08	33.97	107.79
22	2014	4	5	20	72.93	NaN	42.81	70.99
23	2014	4	5	21	59.37	NaN	46.87	46.06
24	2014	4	5	22	55.5	NaN	NaN	28.27
25	2014	4	5	23	59.37	NaN	38.75	18.76

US_Mission China Dataset

This dataset provides PM2.5 of five major cities in China, Beijing, Shanghai, Guangzhou, Chengdu, Shenyang, where five US embassies are located over a long period. The length of the time series vary for the five cities:

- Beijing: 2008-2016
- Shanghai and Guangzhou: 2011-2016
- Chengdu: 2012-2016
- Shenyang: 2014-2016

This is also a very interesting dataset because it potentially can show us the changes over time and patterns of PM2.5 over time so we can find out the reasons for the toxic haze we observed.

Problems:

- Length of time are different for different cities
- Random missing datapoint
- Temporal resolution is unnecessarily high (hourly) for the purpose of this project
- Many files for each city in different time period

Things we did to solve the problems:

- Uniform the data format in terms of time period
- Combine all files into a single csv
- Combine hourly data into daily data using the highest value of the 24 hours within a day

Below is the Python script used for a specific city, i.e. Shanghai:

```
import pandas
import numpy

Shanghai2011 = read_csv('../Shanghai_11_16_hourly/Shanghai_2011_HourlyPM25_created20140423.csv')
Shanghai2012 = read_csv('../Shanghai_11_16_hourly/Shanghai_2012_HourlyPM25_created20140423.csv')
Shanghai2013 = read_csv('../Shanghai_11_16_hourly/Shanghai_2013_HourlyPM25_created20140423.csv')
Shanghai2014 = read_csv('../Shanghai_11_16_hourly/Shanghai_2014_HourlyPM25_created20150203.csv')
Shanghai2015 = read_csv('../Shanghai_11_16_hourly/Shanghai_2015_HourlyPM25_created20160201.csv')
Shanghai2016 = read_csv('../Shanghai_11_16_hourly/Shanghai_2016_HourlyPM25_created20160301.csv')

Shanghai = pandas.concat([Shanghai2011,Shanghai2012, Shanghai2013, Shanghai2014, Shanghai2015, Shanghai2016])
dates = pandas.to_datetime(Shanghai[['Year','Month','Day']])
Shanghai.to_csv("../Shanghai_11_16_hourly/shanghai.csv")
Shanghai['Date'] = dates
max = Shanghai.groupby('Date').max()
max.to_csv("../Shanghai_11_16_hourly/shanghai_daily.csv")
```

Python, pandas, numpy and jupyter

We used pandas and numpy modules of python to do the data transformation.

At last to combine all cities data into a single csv, uniforming the time period length, the following python script was used and we used Jupyter to perform the transformation.

jupyter Untitled-Copy2 Last Checkpoint: 05/09/2016 (unsaved changes)

File Edit View Insert Cell Kernel Help

Code CellToolbar

In [3]:
```
import pandas as pd
import numpy as np
```

In [4]:
```
path = '/Users/Shuai/Dropbox/Vis/finalproject/AirGlobal/Data/US_mission_China'
b = '%s/beijing_daily.csv' % path
c = '%s/chengdu_daily.csv' % path
sha ='%s/shanghai_daily.csv' % path
g ='%s/guangzhou_daily.csv' % path
she ='%s/shenyang_daily.csv' % path
```

In [5]:
```
beijing = pd.read_csv(b)
chengdu = pd.read_csv(c)
guangzhou = pd.read_csv(g)
shanghai = pd.read_csv(sha)
shenyang = pd.read_csv(she)
```

In [6]:
```
bc = beijing.merge(chengdu,how='left', left_on='Date', right_on = 'Date')
```

In [7]:
```
bcg = bc.merge(guangzhou,how='left', left_on='Date', right_on='Date')
```

In [8]:
```
bcgsha = bcg.merge(shanghai,how='left', left_on='Date', right_on='Date')
```

In [9]:
```
bcgshashen = bcgsha.merge(shenyang,how='left', left_on='Date', right_on='Date')
all = bcgshashen.drop('Date',1)
all.columns = ['Beijing','Chengdu','Guangzhou','Shanghai','Shenyang']
all.to_csv("embassy_daily(max).csv")
```

In [10]:
```
all.tail(10)
```

Out[10]:

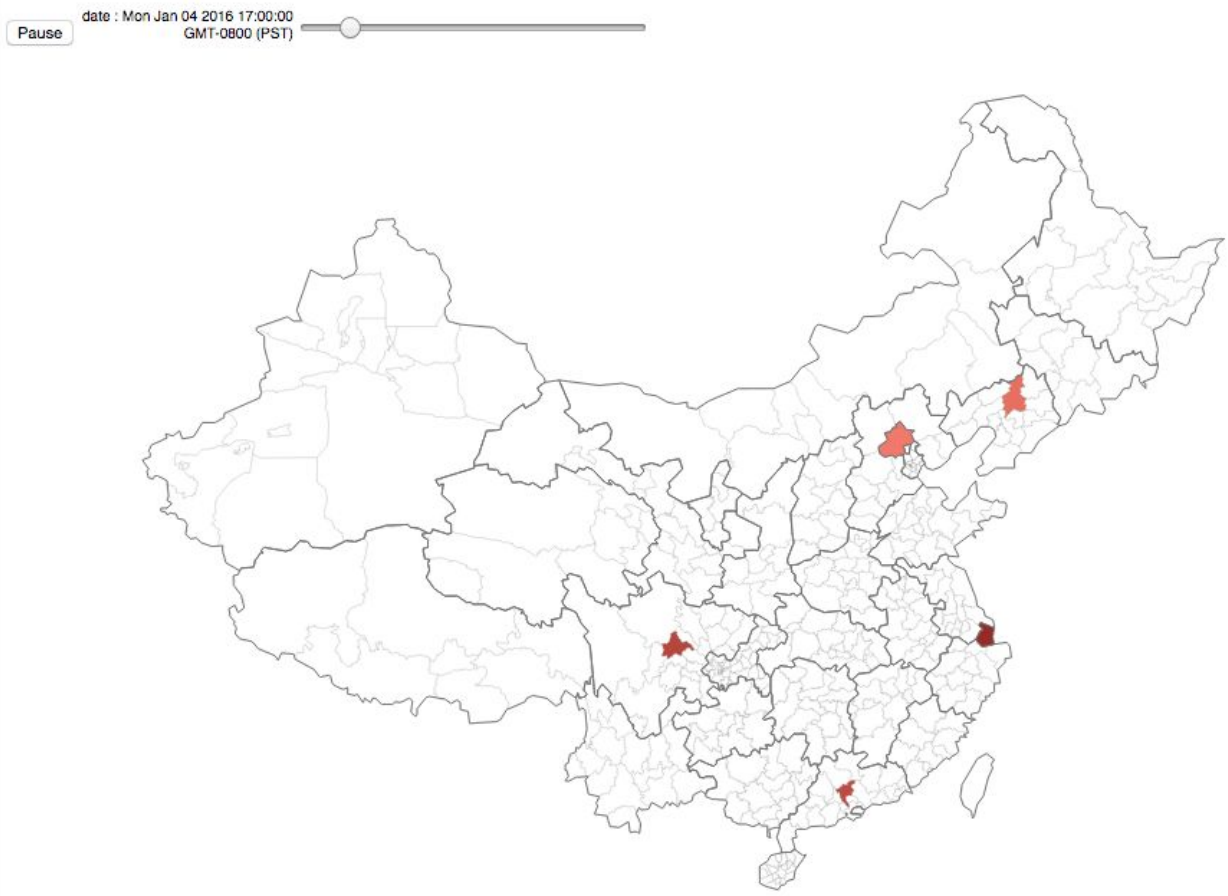
	Beijing	Chengdu	Guangzhou	Shanghai	Shenyang
2845	133.0	78.0	23.0	76.0	145.0
2846	27.0	47.0	45.0	92.0	57.0
2847	26.0	73.0	55.0	67.0	96.0
2848	45.0	80.0	38.0	51.0	227.0
2849	115.0	91.0	39.0	80.0	259.0
2850	309.0	102.0	65.0	54.0	138.0
2851	344.0	141.0	49.0	45.0	223.0
2852	87.0	147.0	29.0	69.0	102.0
2853	115.0	175.0	27.0	65.0	92.0
2854	74.0	117.0	33.0	74.0	232.0

Air Now and EPA Datasets

These two datasets contain global PM2.5 information and are provided in the format of KML(Keywhole Markup Language). This format is designed for Google Earth originally and thus is only suitable for 3D viewers. We learned to understand the data format and extract data from the datasets.

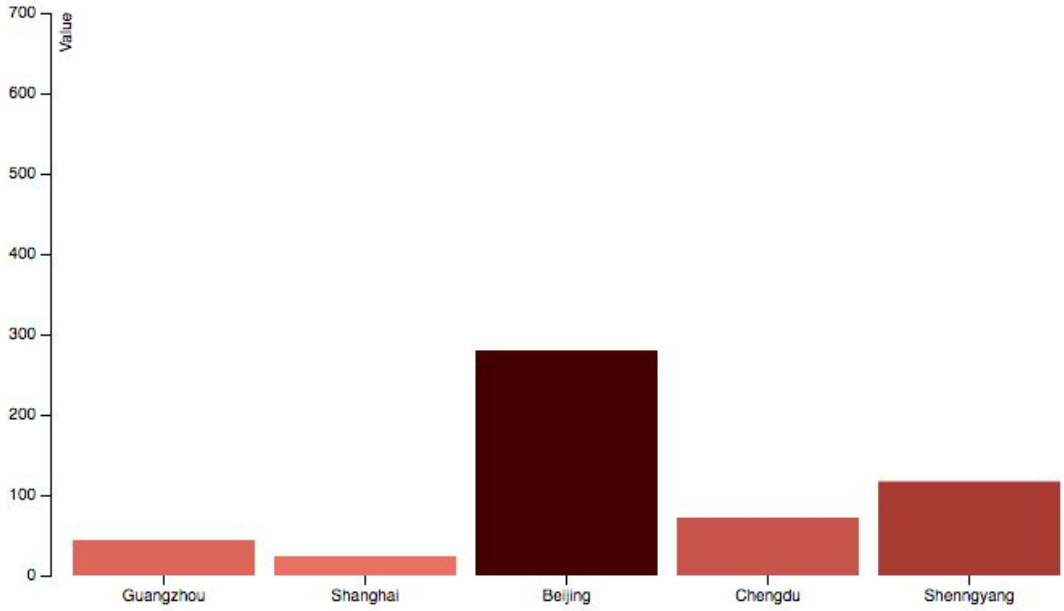
Alpha Release

Chinese map



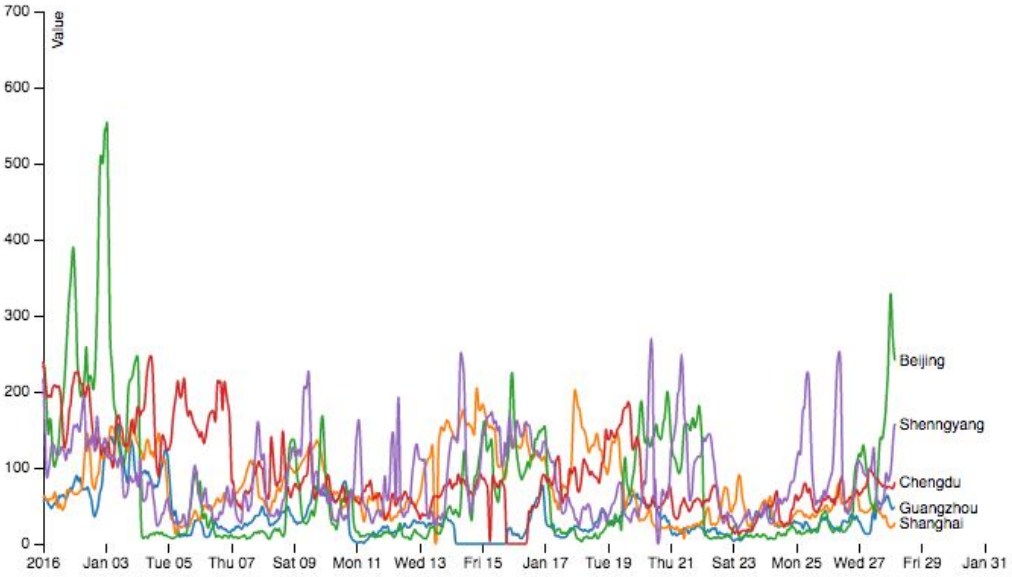
With the embassy data showing five major cities on Chinese map.
Dynamic display over the time-series when you hit “play”.
Use a gradient of red to indicate the concentration of PM2.5

Bar Chart



Use a bar chart to compare PM2.5 among the five cities. The data shown is consistent with the time point at dynamic display. Kind of jumpy visualization.

Line Chart



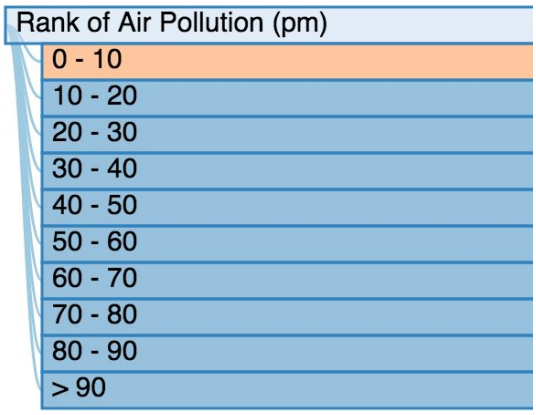
Use dynamic line graph to draw the time series as the data is being “played”.

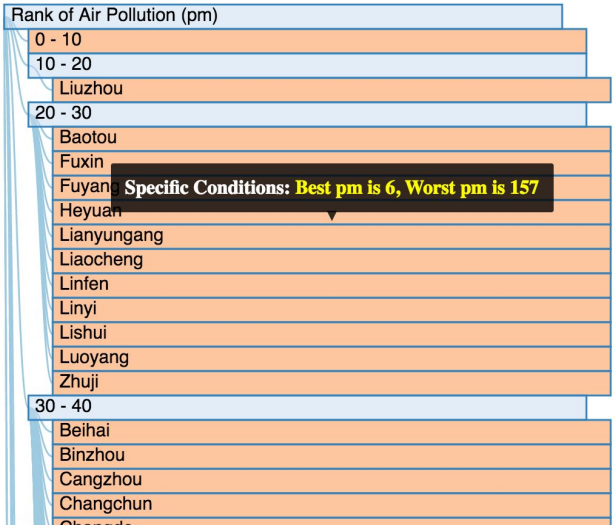
Beta Release

Collapsible Tree Showing Ranking of PM2.5 (Dropped)

This is a collapsible tree. When you execute this program, it first will show you a rank list based on the rank of air pollution.Then, when you click a specific rank item it will unfold to show a sub -list which includes all the cities that belong to this range.

Then, when you move your mouse over the specific city, there will appear a tooltip to show some details such as the best PM2.5 for this city and worst.

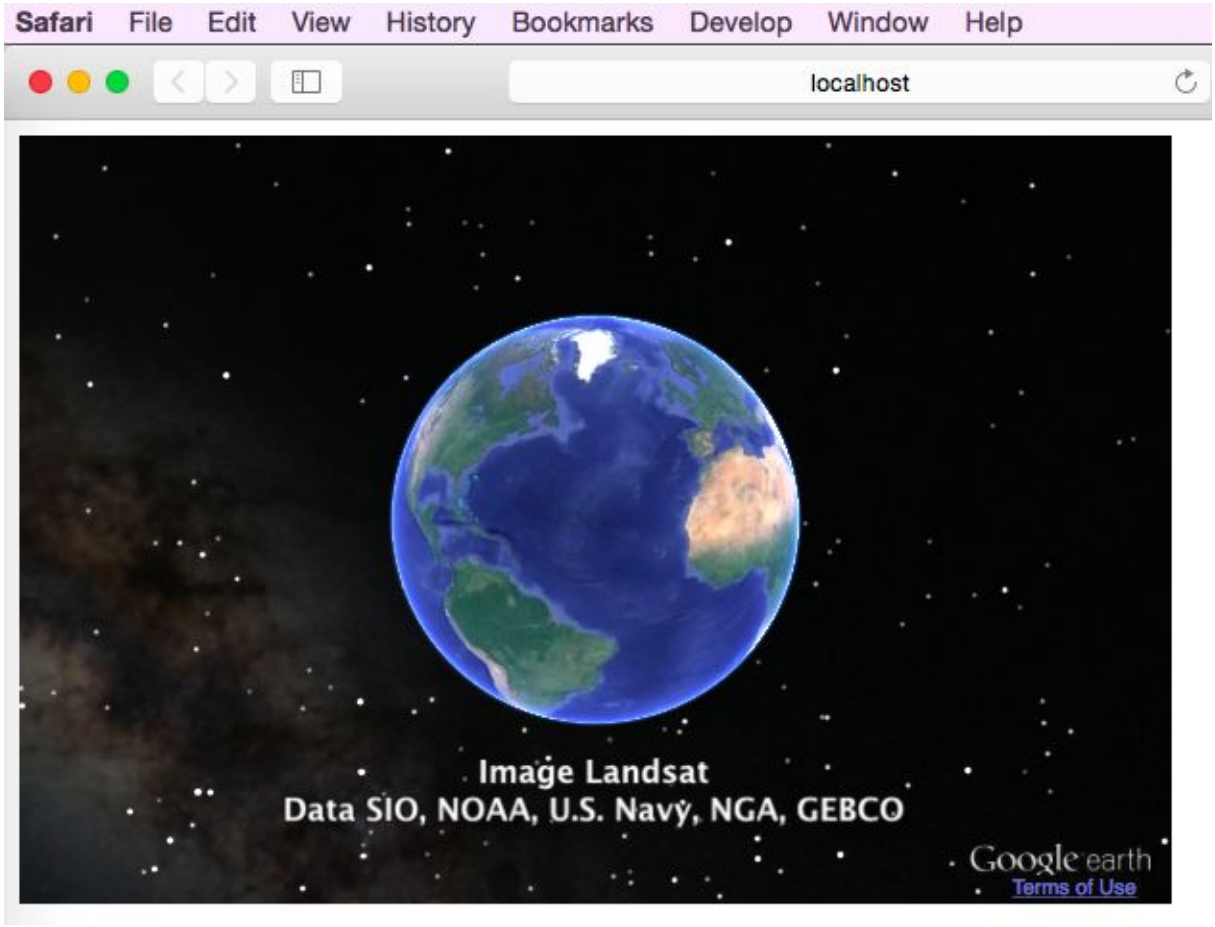




The reason why we do not put this into our project is that there are rank level already on the line graph, also, you can explore details city by city. So, this tree is a little bit repeat of that.

3D Viewer Using Google Earth API (Dropped)

Two of our global datasets are in KML (Keyhole Markup Language) format, which is designed for 3D viewer Google Earth. In order to use the KML global data, we also created a 3D google earth viewer with Google Earth Plug-in, and hoped to use the Google Earth API to visualize the KML data. However there are many issues:



First of all, the KML datasets do not contain the data directly, instead they fetch data through URLs they contained (See figure below). Therefore it relies on all the sites it links to to function properly and it takes a long time to load the data into the viewer.

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://earth.google.com/kml/2.0">
        <Document>
            <name>Air Quality from AirNow</name>
            <open>1</open>
          <Style>
                <ListStyle>
                <listItemType>radioFolder</listItemType>
                </ListStyle>
          </Style>
            <NetworkLink>
                <name>Yesterday's Observed</name>
                <visibility>1</visibility>
                <Link>
                    <href>http://files.airnowtech.org/airnow/today/airnow_yest_obs.kml</href>
                    <refreshMode>onInterval</refreshMode>
                    <refreshInterval>300</refreshInterval>
                    <viewRefreshMode>never</viewRefreshMode>
                    <viewFormat>minx=[bboxWest]&amp;amp;miny=[bboxSouth]&amp;amp;maxx=[bboxEast]&amp;amp;maxy=[bboxNorth]<
                    /viewFormat>
                </Link>
            </NetworkLink>
            <NetworkLink>
                <name>Current Conditions</name>
                <visibility>1</visibility>
                <Link>
                    <href>http://files.airnowtech.org/airnow/today/airnow_conditions.kml</href>
                    <refreshMode>onInterval</refreshMode>
                    <refreshInterval>300</refreshInterval>
                    <viewRefreshMode>never</viewRefreshMode>
                    <viewFormat>minx=[bboxWest]&amp;amp;miny=[bboxSouth]&amp;amp;maxx=[bboxEast]&amp;amp;maxy=[bboxNorth]<
                    /viewFormat>
                </Link>
            </NetworkLink>
```

Secondly, the Google Earth API has been deprecated as of 12/12/2014 and requires an older version of Chrome(5.0-39.0, while most of us have 50.0+ running). Same issue exists for Firefox. The visualization we created can only be viewed on Safari. Therefore, considering the complexity of the two Chinese datasets, we decided to drop the two global datasets despite the effort in processing the data.

Google Earth API Developer's Guide

☆☆☆☆☆

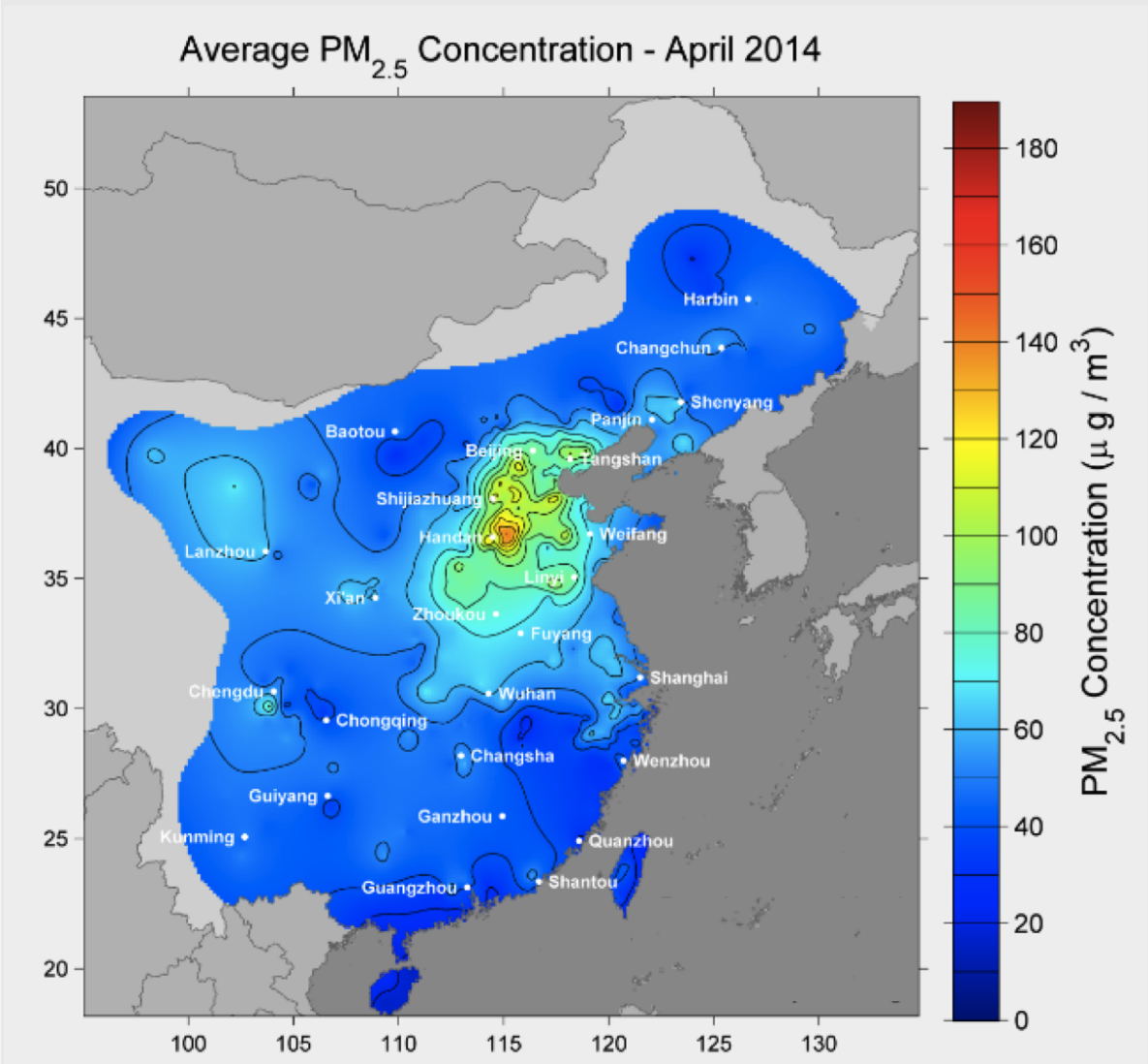
Note: The Google Earth API has been deprecated as of December 12th, 2014. The API will shut down by end of 2016, and will continue to work on supported browsers until that date.

The Google Earth Plug-in is currently supported on the following platforms:

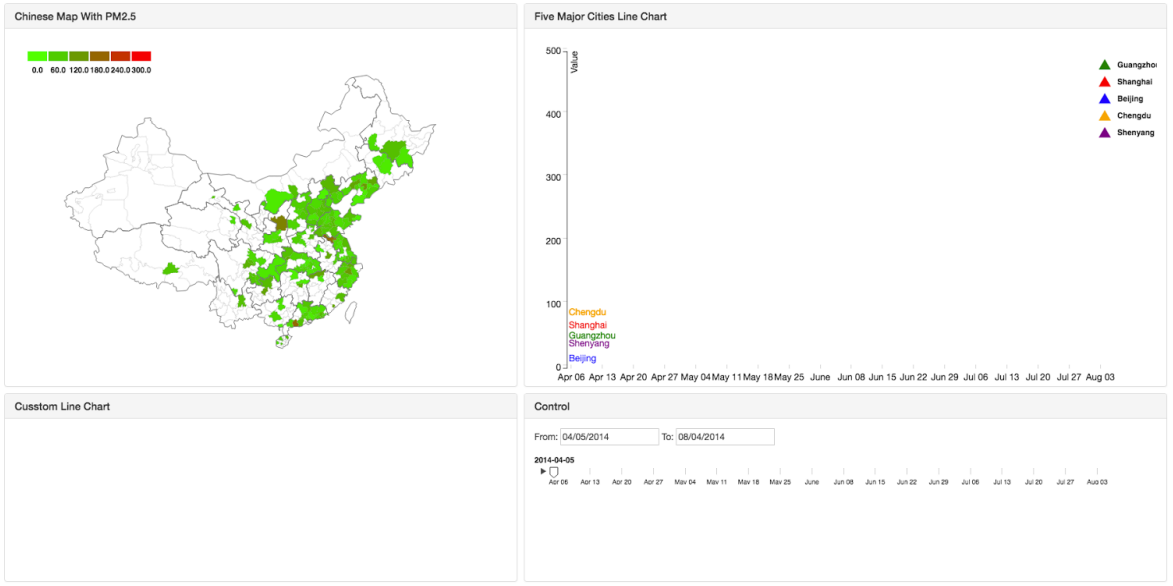
- **Microsoft Windows (Windows XP, Windows Vista, Windows 7, and Windows 8)**
  - Google Chrome 5.0-39.0[1] (32-bit)
  - Internet Explorer 7-9, and 10-11 with Compatibility View (32-bit)
    (Note that the Windows 8 browsing mode with Internet Explorer does not support plugins.)
  - Firefox 11.0-34.0[2]
    (The plug-in won't install while Firefox is running.)
- **Apple Mac OS X 10.6 or later (any Intel Mac)**
  - Google Chrome 5.0-39.0[1]
  - Safari 3.1+
  - Firefox 11.0-34.0[2]
    (The plug-in won't install while Firefox is running.)

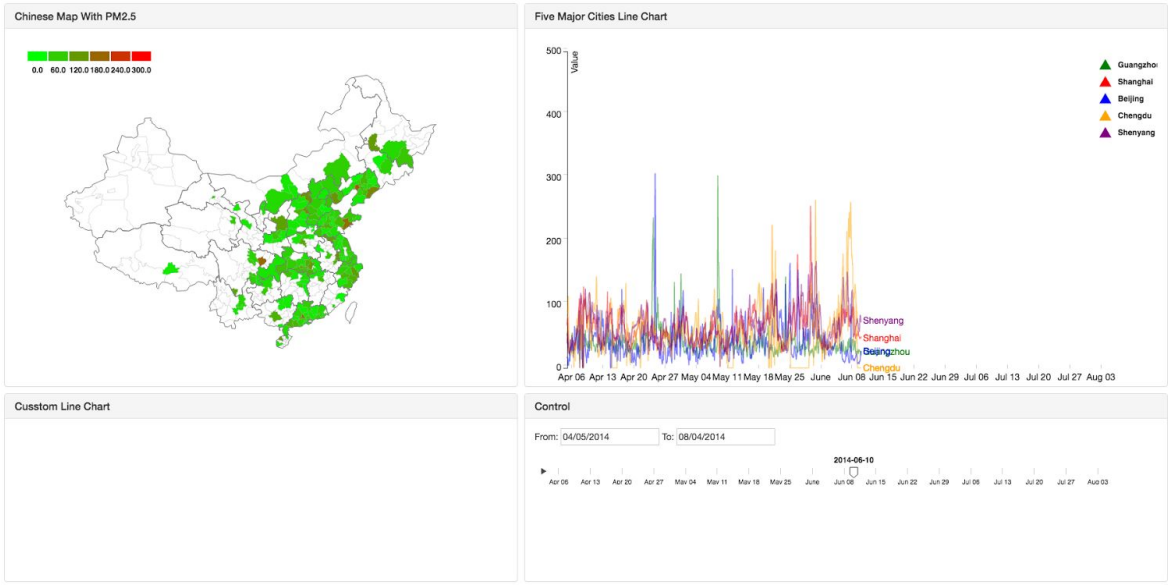
Color Gradient 2D Map with Interpolated Data(Dropped)

We proposed to provide color gradient 2D map with Interpolated Data using the Berkeley dataset. And we successfully generated the interpolated gridded data according to the algorithm described in the original paper(see reference in project proposal). However, we realized that it does not add much information to the Chinese map visualization, and in fact people around us thinks it is more interesting to know the air pollution condition for particular cities as very few people know much about the regions of China. And since we already have a full web page with important visualization and interaction tools, we decided to drop it to save space. See below for the prototype we got.



Dynamic Line Graph with Selection of Time Period





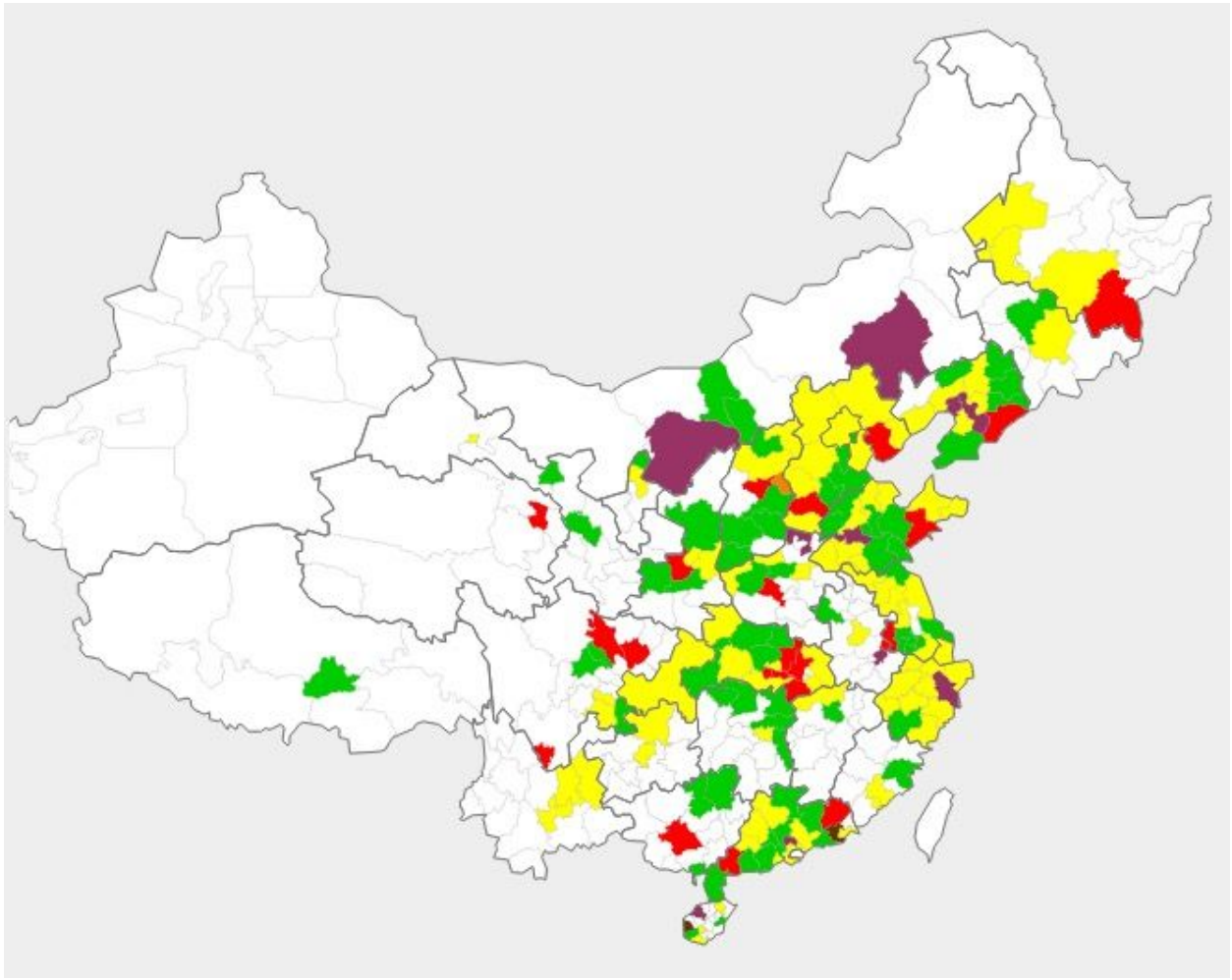
After all the previous exploration and decisions we decided to keep the dynamic line graph. We added interactions to allow users to select cities and date range. We also changed the color scheme to better show the severity of PM2.5 pollution.

Final Release Features

Intro Page

A static page with introduction to the problem, sample pictures of the toxic haze caused by high concentration of PM2.5 in three major cities in China, as well as the color scheme of Air Quality Index according to WHO standard, explaining how hazardous PM2.5 is at different concentrations.

Map of Chinese Cities

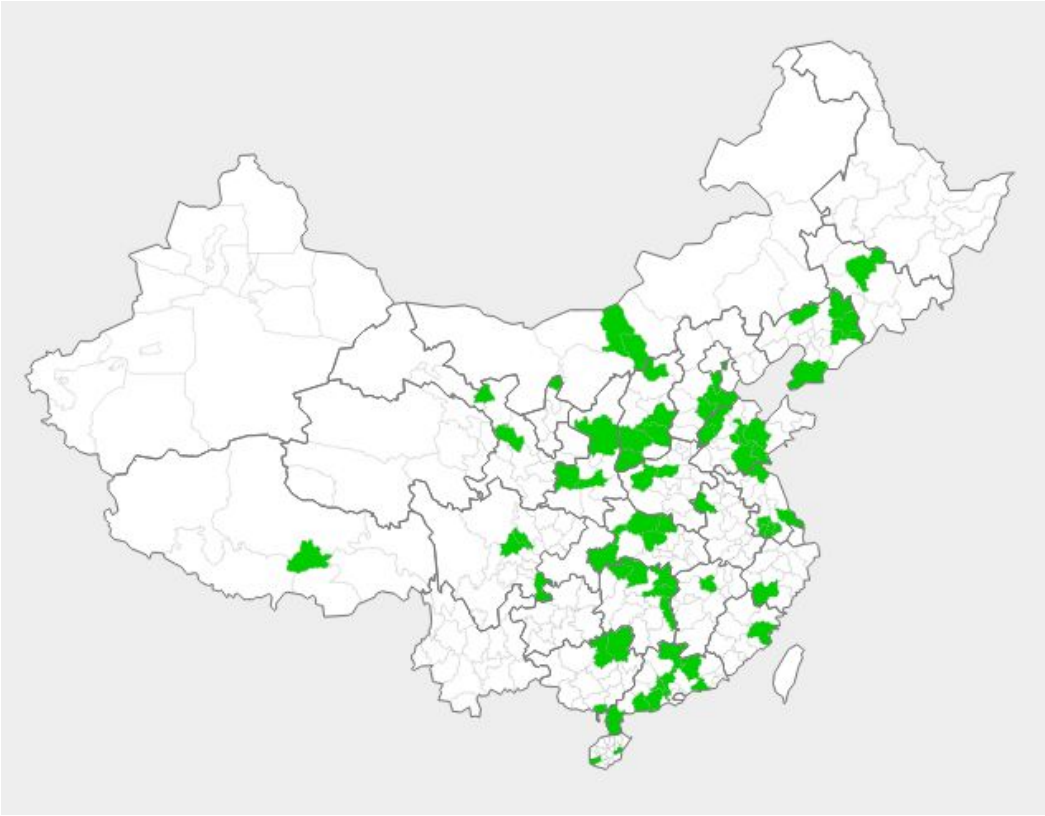


We use geojson to generate the chinese map. Base on the data, we translate the PM2.5 into AQI color and fill it to each city. This way you can easily see how severe the pollution is in that city.

Also, the map is showing the same cities as the small multiples. After search or filter, it will show only the cities of the user's interest, helping the user to locate the city geographically on the map.

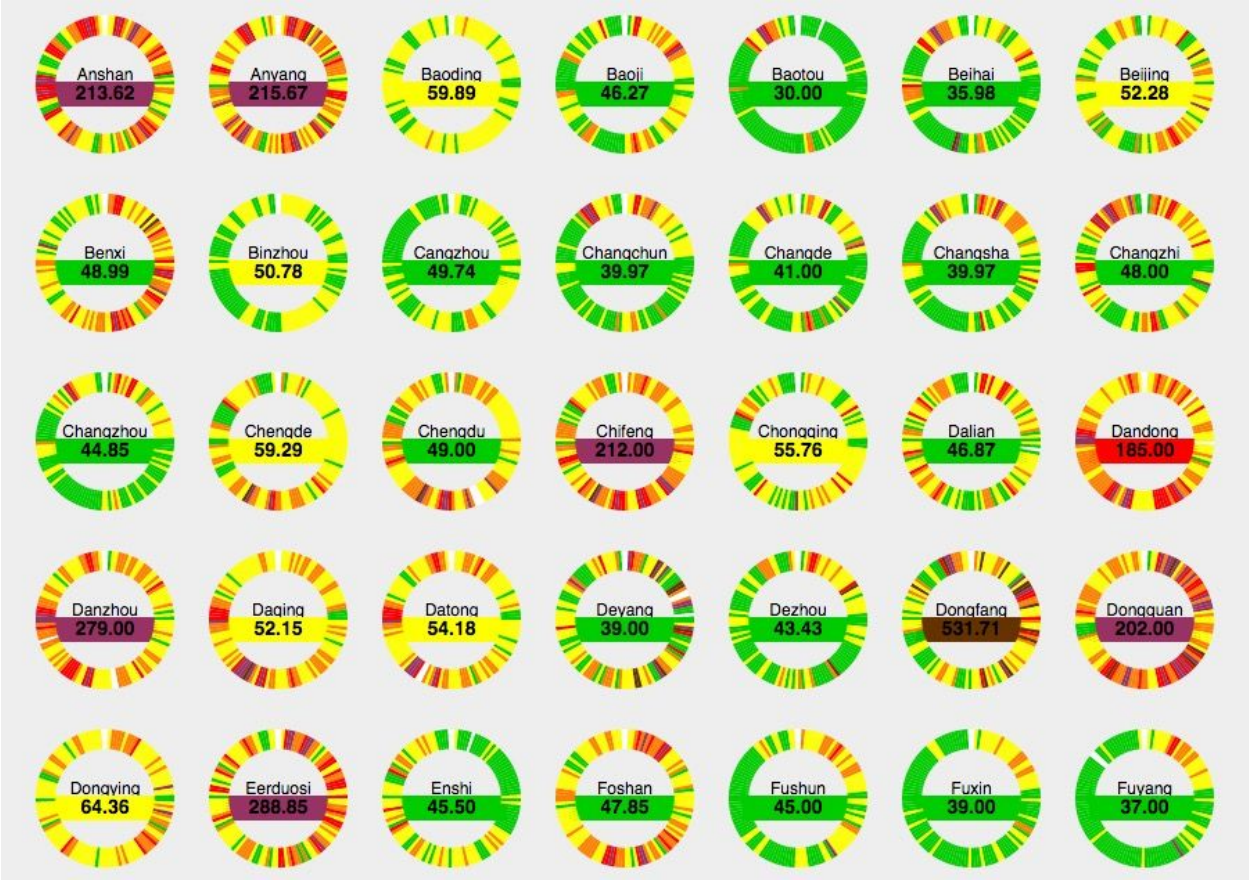
The map also provides tooltip showing the name of the city.
By clicking on the cities on the map, the user can add/remove city data in the line graph as well, allowing the user to check the time-series data more closely.

With search(Guangzhou,Shanghai) in chinese map



With filter(0 ~ 50) in chinese map

Small Multiples

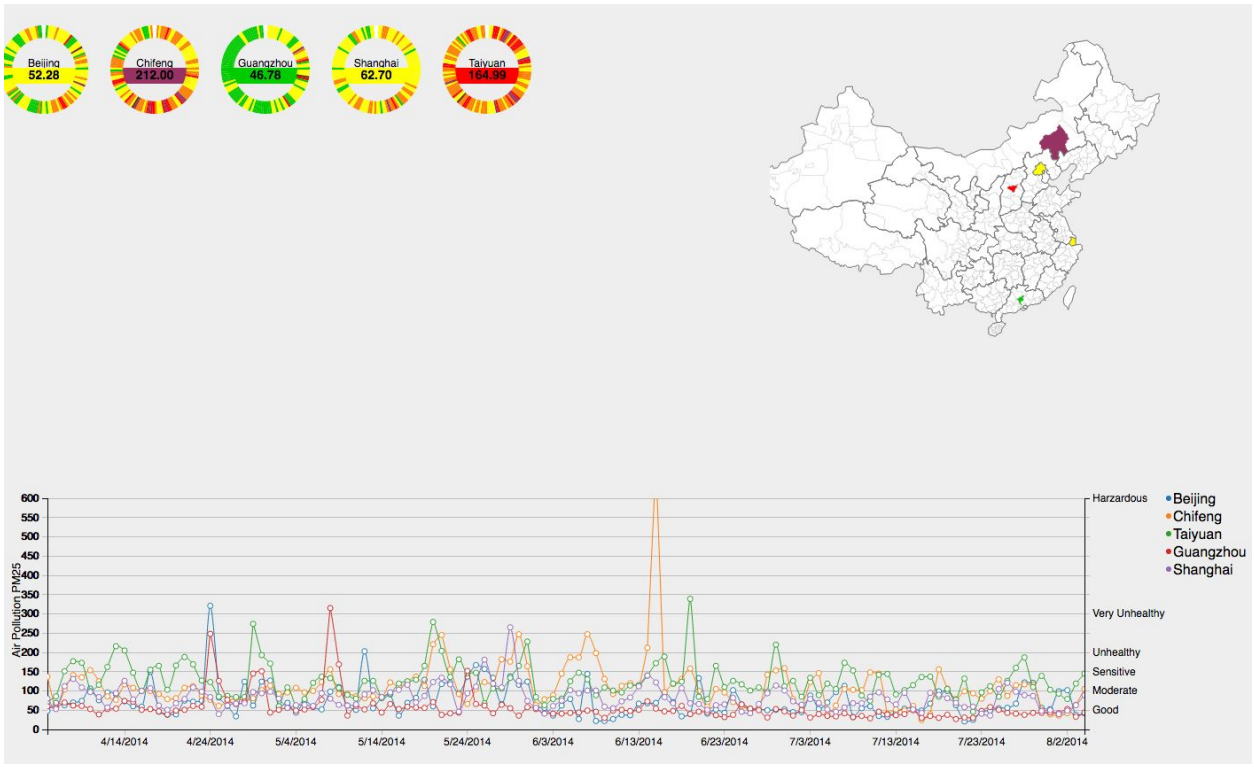


We use circular graph to implement our small multiples. Base on the “from” date and “end” date, we get the data from our data model API and evenly divide a circle into stripes base on the number of days (by default it takes the entire time-series dataset). After that, each stripe will be color coded based on the PM2.5 color scheme. The name of the city and the median(by default) value of PM2.5 concentration are shown in the center bar of the circle.

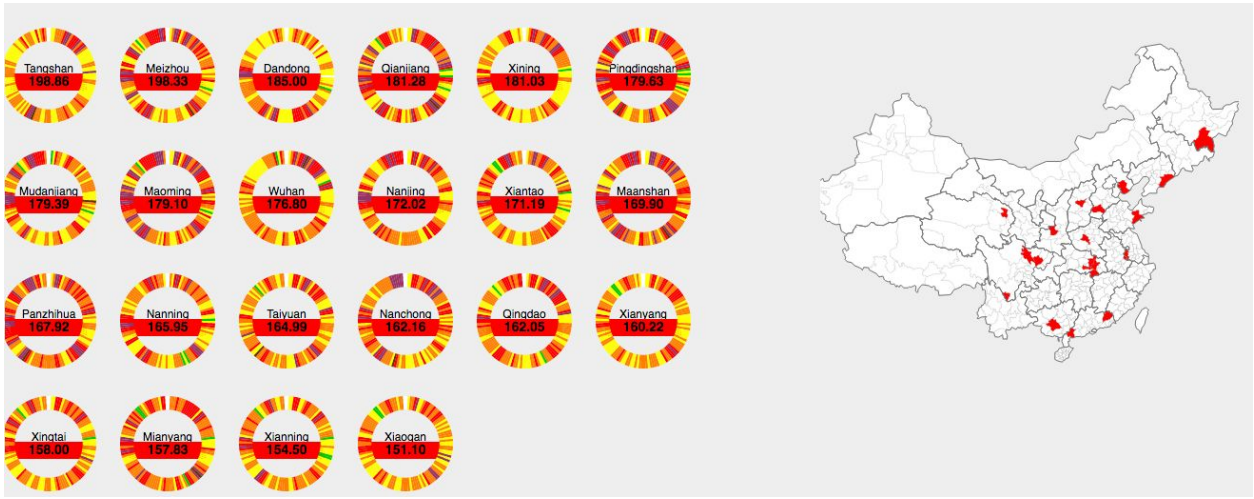
This visualization allows the user to first get an overall idea of the pollution of the city within the period of interest, and to compare the conditions among different cities very easily.

We also provide a variety of user interactions through small multiples. When you move the mouse to a particular stripe on any circle, all the circles will change to show the data of that particular date in the center bar, together with the color encoding, one can easily compare the pollution on a particular date based on his/her observation across a bunch of cities of interest.

The small multiples respond to search, select dates, sorting and filtering in the navigation bar, it coordinates with the map to show the location of the cities, by clicking on a particular circle you can also add/remove line graph in the panel below.



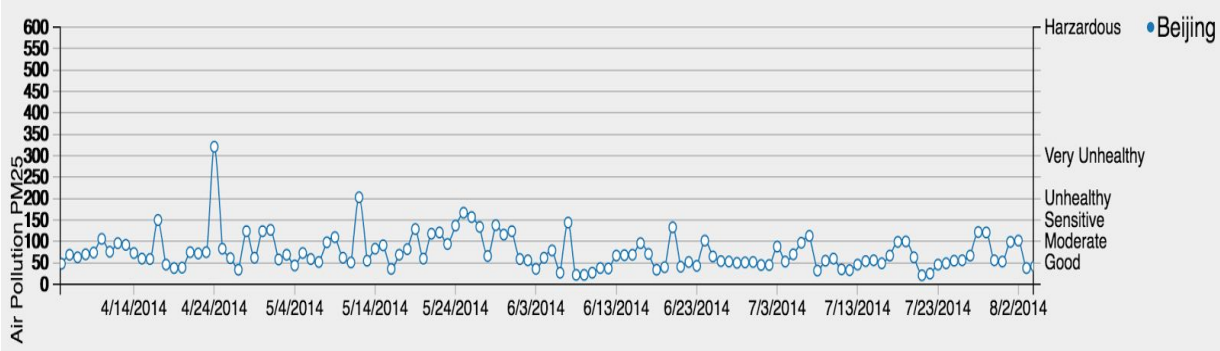
Search: Beijing,Shanghai,Chifeng,Taiyuan,Guangdong
Click the small multiples or the highlighted cities in the map to add the line graph



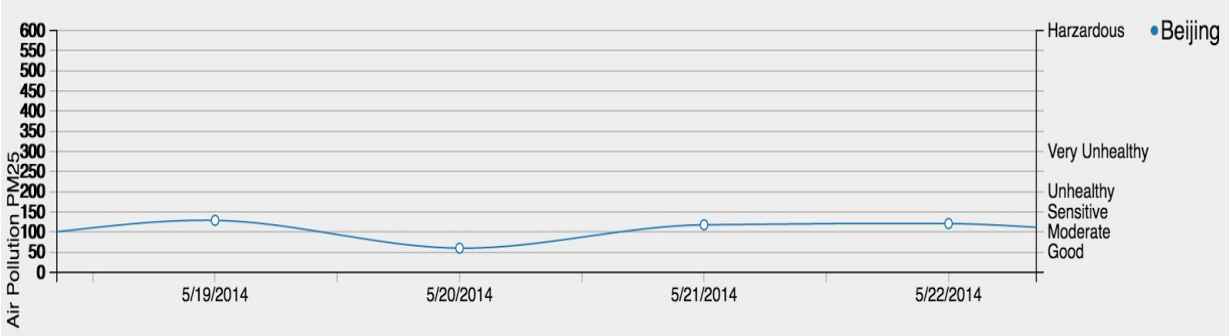
Sort: by Median
Filter: 151-200 (the unhealthy group)

Line Graph

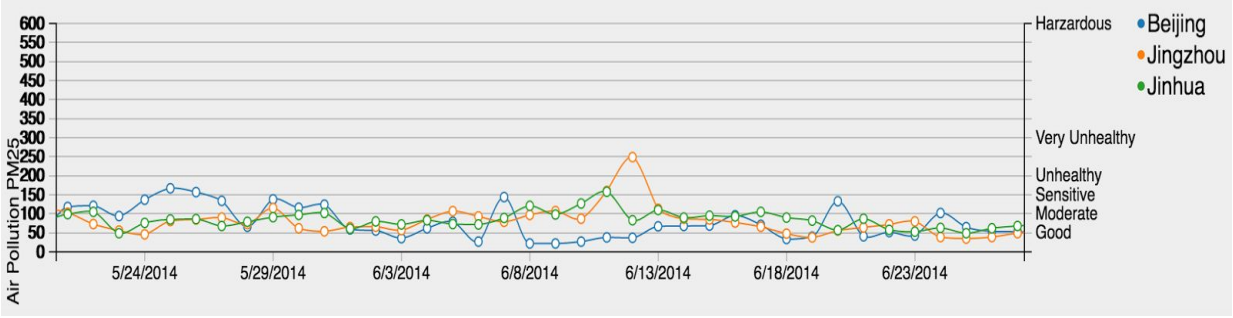
The x-axis represents the range of date. The left y-axis represents the range of the air pollution. The right y-axis represents the rank of the air pollution such as good, moderate and so on. And at the rightmost, there is the list of chosen cities for comparison.



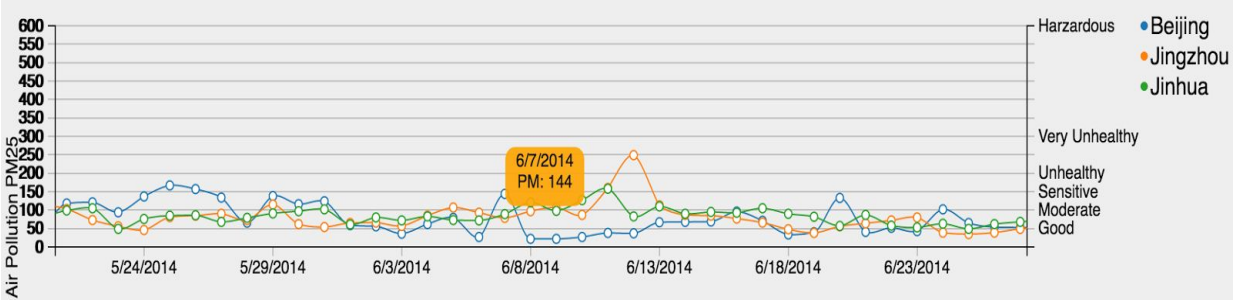
As you can see from below figure, you can zoom in or out this line graph based on a specific date or drag around.



As you can see, you can add cities as you want. Then all the chosen cities will appear at the rightmost with different colors.



Last point is that when you move mouse over to a specific point, there will be a tooltip to show some details such as specific date and value of air pollution.



Navigation Bar

The navigation bar have 6 functions.

1. The link to the intro page.

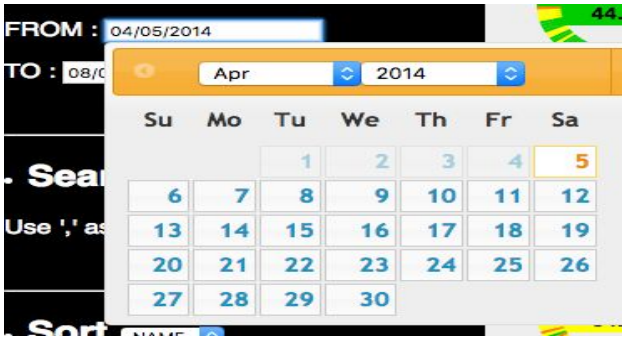
   When you click the INTRO, the browser will open a new tab to the intro page.

2. Change data source.

   There will be two data sources, one is Berkely and the other one is Embassy.

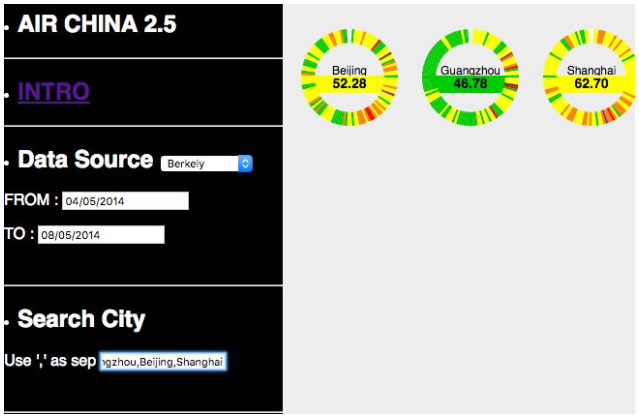
3. Change the from and to date.

   When you click the input of the FROM and TO, it will trigger an UI widget to let you select the date.



4. Search by city name.

   You can type the city name you want to search in the search input. If you want to search multiple cities, you can use the ‘,’ as the separator.



5. Sort by different attributes.

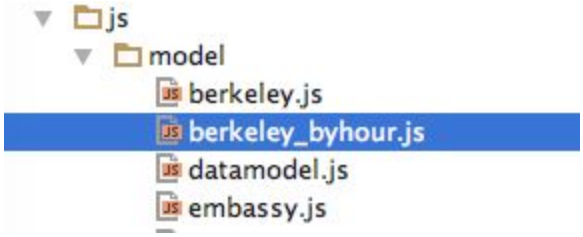
   There are 4 kinds of attributes to sort, name, max, min and med.

6. Filter by the value range.

   The filter of the value range based on the AQI values standard.

Data Model API

We have multiple very large and complicated datasets for a light-weight web application. In order to make the application efficiently responsive during user interaction and easy to use for the visualization, we built data models and provide APIs that allow the visualization part of the application to get data,statistics and computated result fast.



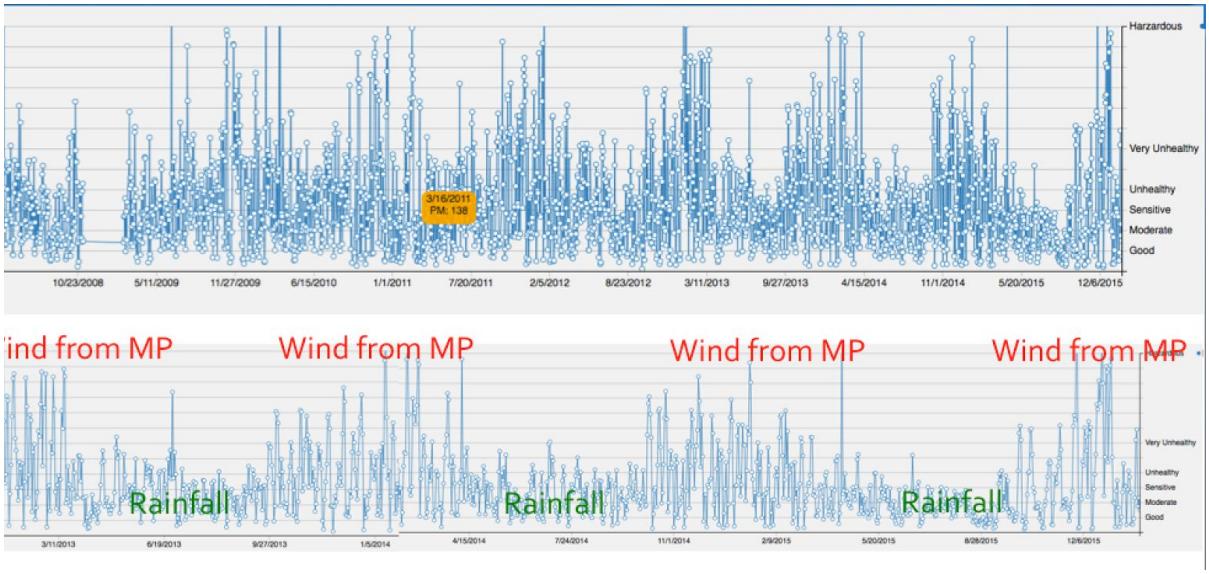

```
Console.prototype = {
    init : function () {
        var root, uschinaContainer, options = {},

            root = new Backbone.View({el: $('#ag-root')});
        options.rootView = root;
        options.data = {};

        options.data.berkeley = new Berkeley();
        var defer_berkeley = options.data.berkeley.load();
        options.data.embassy = new Embassy();
        var defer_embassy = options.data.embassy.load();

    $.when(defer_berkeley, defer_embassy).done(function () {
        uschinaContainer = new USChina({root : options.rootView, data : options.data})
        uschinaContainer.render();
    });
```

We built two data models Berkeley and Embassy which subclassing the common parent class Datamodel. When the server is initiated, we initialize the two models by calling the load() function which loads individual csv files and create data structures that are easy to be indexed and computed in memory. For each model set we have a map to map city name with city id on the map. Each model has two arrays of arrays: one is an array of time points, each element being an array of all city values at that time point; the other is an array of all cities, each element beijing an array of all time points for that city. We also keep a map that match city name and their index within the arrays. This way we can fetch data for a particular set of cities at particular time points very fast. Below are the APIs we provide for the visualization to fetch data efficiently during user interaction:

getOneCityInTimeRange() getAllCityAtDay() getAllDayForCity()
getTimeRangeData() getCitiesData() getCitiesTimepoints()
getDayIndex() getMicroseconds() getCityIndex()
getMapID() getAllCities()
getAQI() getColor()

Explore the Data with the App

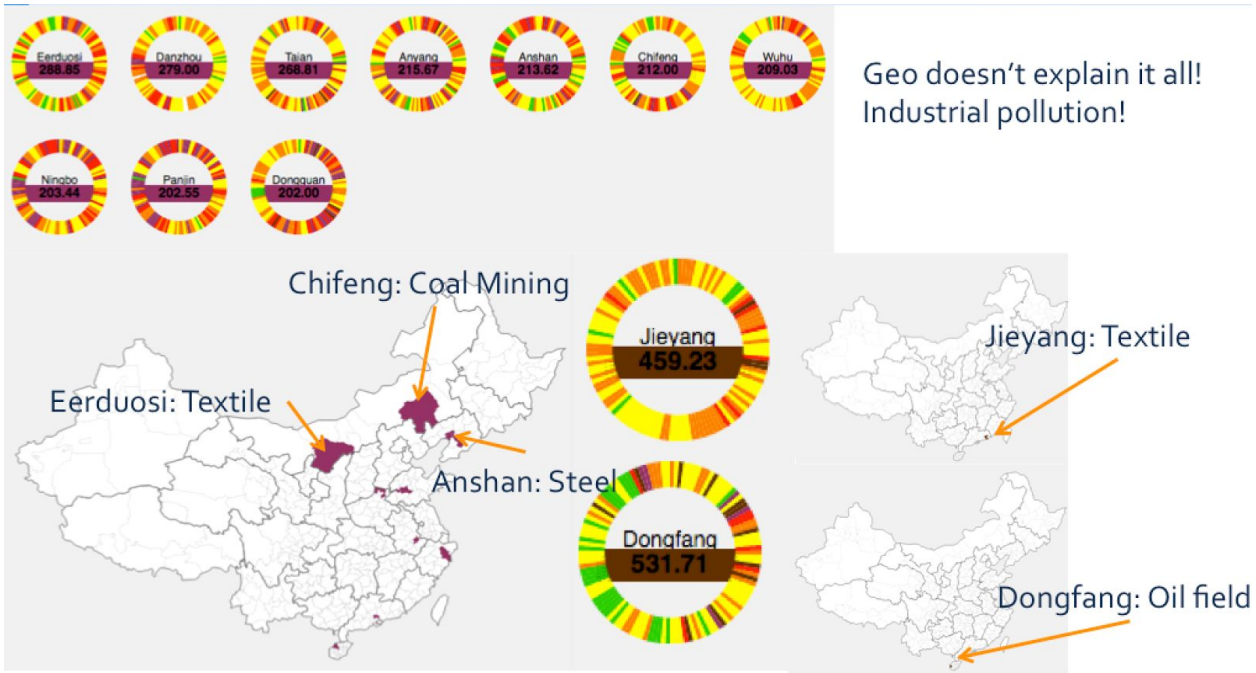
Natural factor



Conclusion:

Natural factor: seasonal, rainfall, wind from Mongolian Plateau

Industry factor

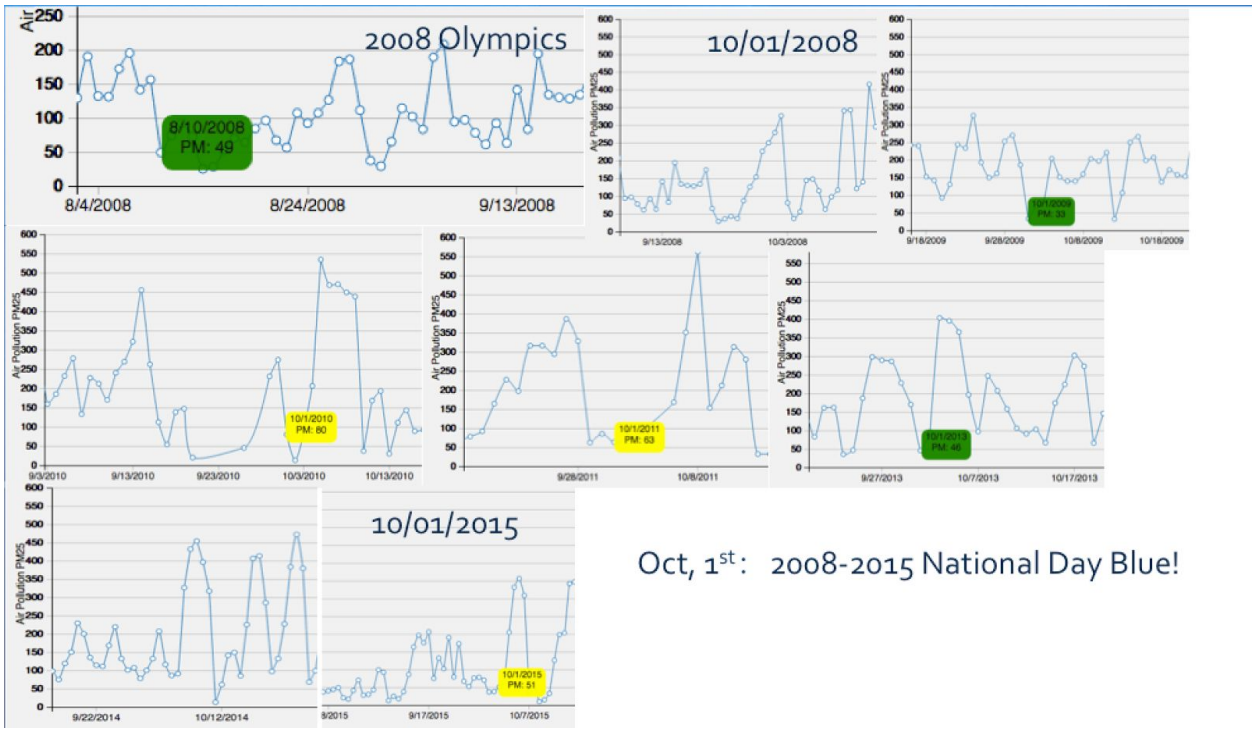


Conclusion:

Industrial factor:

- Coal mining,
- Steel,
- Textile,
- Oil.

Human factor



Conclusion:

Human activity/Government Interference:

- Artificial rainfall,
- Temporal pause of polluting industry,
- Temporal vehicle regulation

Summary of Key Features

Two datasets:

- Berkeley Earth
- US Embassy

Visualization:

- Geo-visualization
- Line graph
- Small multiples

Interaction:

- Tooltip
- Zoom in/out
- Panning
- Ranking/sorting
- Filtering
- Statistics(MIN, MAX, MEDIAN)
- Select city
- Select time period