HW4

Huajun CHai

998584945

Traffic Data analysis using google maps and shiny interface

Introduction

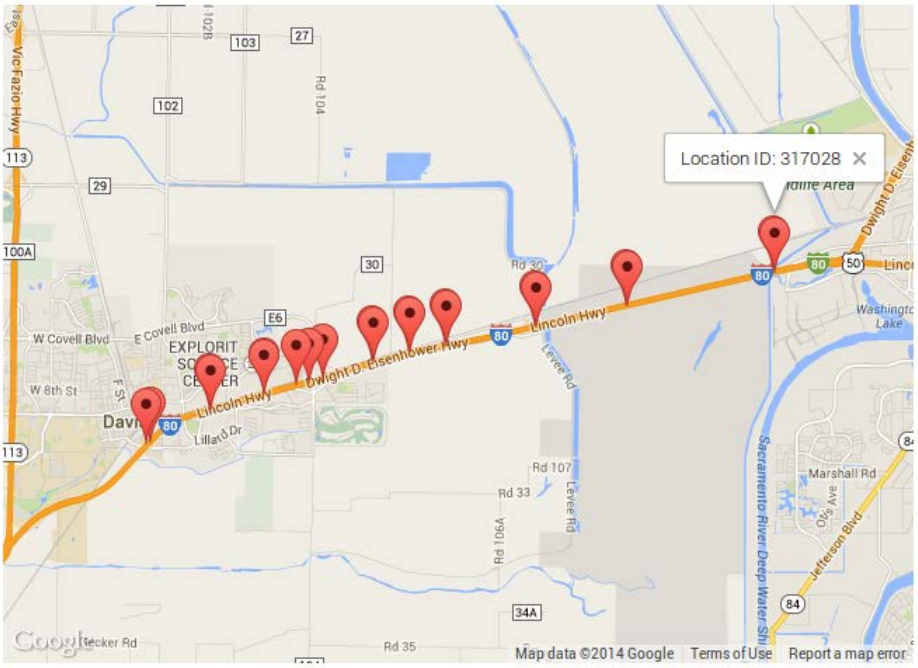
In this homework, I am trying to visualize the traffic data detected by the sensors located in the highway network in California. Actually, there is a large project named "PeMS" which is a project aiming to collect real-time traffic data within the road network. The traffic information PeMS can collect include traffic speed, traffic volume, traffic density and so on. The data is available online from the PeMS website and a full description of PeMS project can also be found on that website[1].

In transportation research, there are some basic but important variables and relations. With some visualization and regression tools we are able to get these variables and relations which can be used in the further research.

Set up of the structure

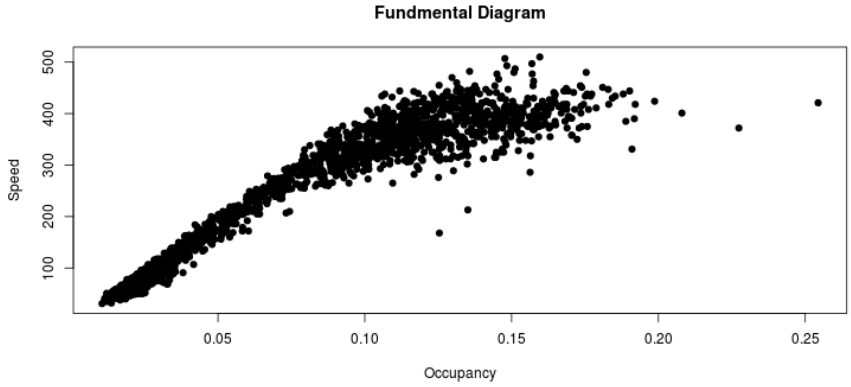
Geometrical visualization

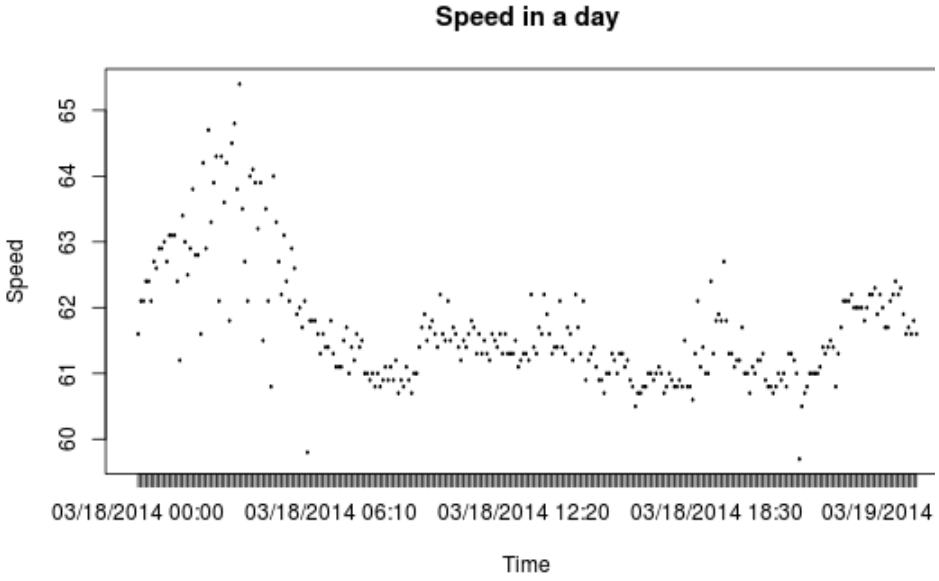
In the middle to the left, there is a google map showing all the detector stations in the network. They are shown as markers which can be clicked. Once put your mouse on the marker, the id number of the station will show up.

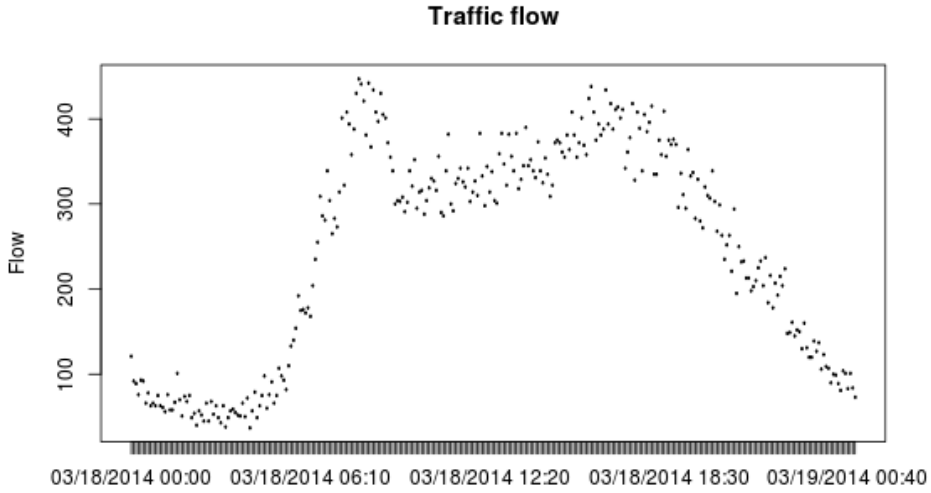


Data visulaztion

In the right side of the page, data is plotted. One is for fundamental diagram, the other two are speed and traffic flow profile over one day.







How to set up the platform

Tools used

Shiny: shiny is server implementation in R. It consists of a "server.R" and a "ui.R".
"server.R" serves as a virtual server, which will handle and process the requests from the webpage.
"ui.R" takes care of the webpage display and user interface thing. You can configure the layout og your webpage in this file. However, in my implementation, I use a "html" file, "index.html", to design the layout which is equivalent to that of "ui.R".

Google map API: in order to display the stations geometrically, we need the interface of google map or any other online map application, such as open maps. Google map API is a html language, which is integrated in the "index.html" file.

One thing is to add markers corresponding to the coordinates of the stations. This is quite easy since the is a "marker" object in google map API. We can simply add markers to the map canvas.

The other thing is to add reactive actions to the map. As long as user clicks the right button or marker, the display, i.e. the plots will change.

- google.maps.event.addListener(marker, 'click', function() {
- infowindow.setContent(this.title);
- infowindow.open(map, this);
- }); // This function will show the Station ID once the user click the // marker
- google.maps.event.addListener(marker, 'click', (function(i) {
- return function() {
- SetValue(i+1);
- }
- })(i)); // If user clicks the marker, the ID entry will be updated. And once the ID is updated, "server.R" receives the signal, and will flush the plots.

- R scripts
  - In order to get the desired data from the PeMS website, some R scripts are used to obtain that.
  - "curl" is used to download data. Since PeMS requires user to login before downloading any data, so we need "cookie" to get "curl" to work properly.
    - curl -L --cookie "COOKIE" "URL" > data.txt
- XML

- json: a json file containing geo-location information is used in the "html" file to store the coordinates of the detectors.

```
var Stations = [
[-121.735229,38.54053, "318102"],
[-121.73425,38.541047, "318113"],
[-121.71973,38.547128, "316773"],
[-121.719907,38.547332, "316778"],
[-121.706428,38.550149, "318017"],
[-121.706609,38.550321, "318022"],
[-121.698617,38.552058, "313979"],
[-121.695688,38.552241, "314013"],
[-121.692178,38.553123, "314025"],
[-121.679867,38.556203, "316783"],
[-121.680098,38.556385, "316788"],
[-121.670963,38.558143, "318067"],
[-121.671206,38.558305, "318068"],
[-121.661873,38.55951, "318076"],
[-121.662131,38.559664, "318077"],
[-121.640097,38.563017, "318053"],
[-121.640398,38.563162, "318052"],
[-121.617723,38.567081, "316803"],
[-121.61782,38.567247, "316808"],
[-121.581706,38.573494, "317028"],
[-121.581761,38.573676, "317033"],
]
```

- I place the json file in a public folder in dropbox. The file can be updated dynamically, using the function in "scripts.R".

Before you run

Before you can finally run the program, you need to install the "shiny" package in R to enable the shiny function.

- install.package(shiny)
- library(shiny)

After all the required libraries are install, you can easily run the program by:

- runApp("shinyapp")

One thing to notice, for shiny applications, there should be a folder containing "server.R" and "ui.R". The structure of it is shown as follows:

```
~/shinyapp
|-- ui.R
|-- server.R
```

Future functionalities to be added

Now, the plot is static. It doesn't allow user to choose the options or change the time window. In the future, we'd like to add these functionalities in the webpage.

Since we are analyzing the traffic data, it will be useful and meaningful to export and save the results to some files, locally or online. Also, it will be good if user can choose what information to export.

[1] http://pems.dot.ca.gov/