ECE 232E Project 5

Graph Algorithms

Jui Chang
Wenyang Zhu
Xiaohan Wang
Yang Tang

1 Stock Market

Question 1.

(1) In order to know the range of ρ_{ij}, we can consider the questions in the below two special situations.

When $r_i(t)$ and $r_j(t)$ are independent, then $< r_i(t)r_j(t) >=< r_i(t) >< r_j(t) >$. Therefore, the nominator of ρ_{ij} becomes 0. So, we get $\rho_{ij} = 0$. On the other side, when $r_i(t)$ and $r_j(t)$ are corelated, then we suppose $r_i(t) = k * r_j(t)$. Therefore, $< r_i(t)r_j(t) > = k < r_i(t)^2 >$ and $< r_i(t) >< r_j(t) > = k < r_i(t) >^2$. Besides, $(< r_i(t)^2 > - < r_i(t) >^2)\left(< r_j(t)^2 > - < r_j(t) >^2\right) = k^2(< r_i(t)^2 > -k < r_i(t) >^2)^2$. So, the final result of $\rho_{ij} = \frac{k}{\sqrt{k^2}}$. When k is positive the value is equal to 1, and when k is negative the value is equal to -1. Hence, we get $\rho_{ij} = \pm 1$.

Therefore, the upper bound of ρ_{ij} is 1 and the lower bound of ρ_{ij} is -1.

(2) In order to provide a standard for weight, the correlation between stock i and j should be based on the normalized factors. For q(t)'s, we can find their expectations E[q(t)]'s are not equal, therefore not providing that standard for each i. However, for r(t), we can easily prove that the expectations are all 0, i.e. $E[r_i(t)] = \log(1) = 0$. Therefore, we use log-normalized instead of regular return.

Question 2.

With the expression $w_{ij} = \sqrt{2(1 - \rho_{ij})}$ and the expressions in the first question, we can compute the edge weights. Then, we plot the degree distribution of the correlation graph and the histogram for the un-normalized distribution of edge weights.

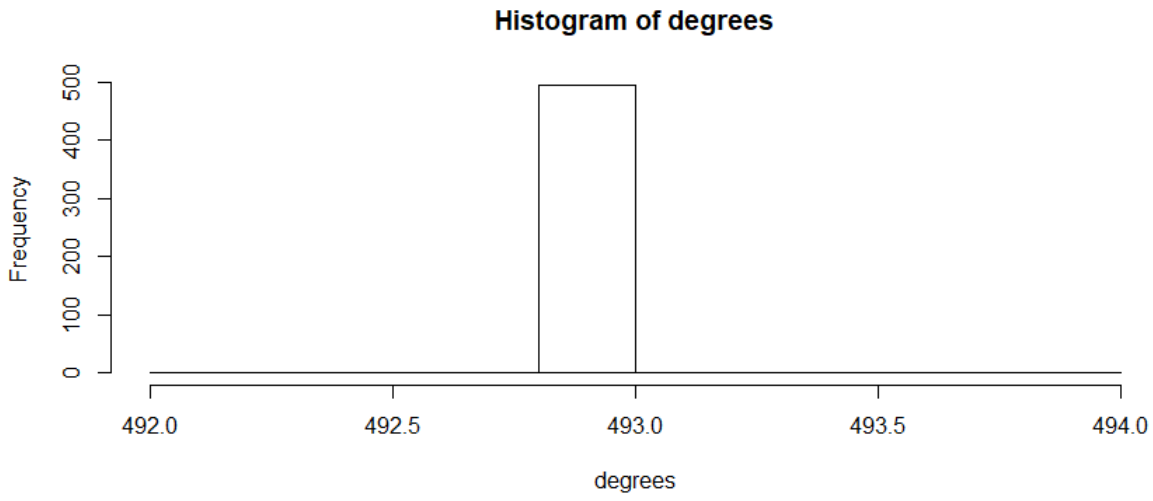


Figure 1. The degree distribution of the correlation graph

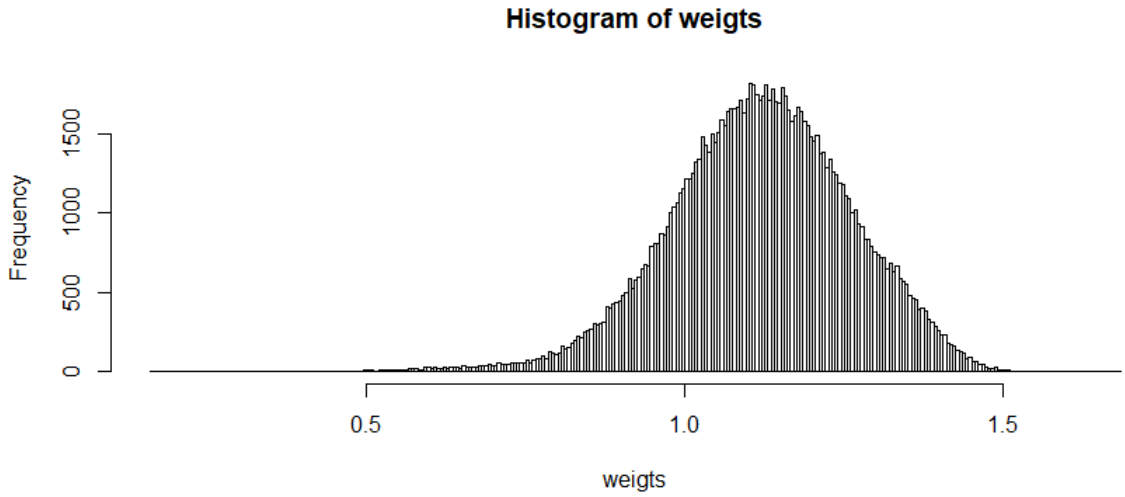


Figure 2. The histogram of un-normalized distribution of edge weights

Question 3.

For the correlation graph, its minimum spanning tree (MST) based on sections (which can be found in Name_sector.csv) is shown below with color-coding.

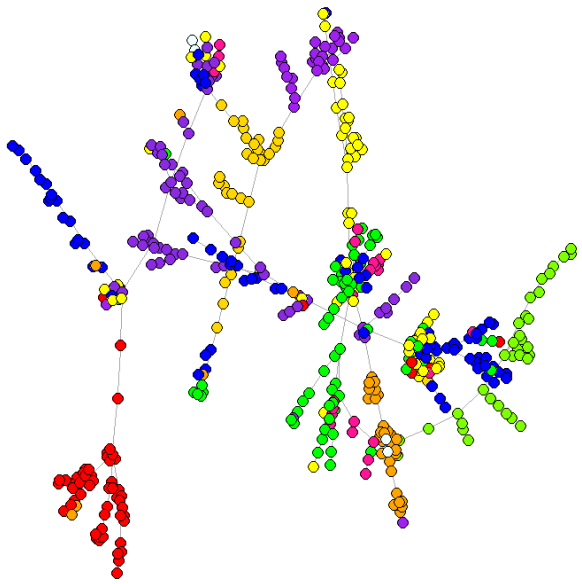


Figure 3. MST of the correlation graph

Through observation, we can find the pattern in MST. The nodes with the same sector are more likely to be connected in MST while the ones with different sectors are less likely to be connected. This means that edges between the nodes with the same sectors have lower weights than the ones between the nodes with difference sections. Hence, a node has higher correlation with the nodes in the same sectors, while it has lower correlation with the nodes in the different sectors.

Question 4.

In order to finish the task of predicting market sector of an unknown stock, two different methods of evaluation are defined here. The values for each method are calculated:

When $P(v_i \in S_i) = \frac{|Q_i|}{|N_i|}$, α =0.112

When $P(v_i \in S_i) = \frac{|S_i|}{|V|}$, α =0.114

In the first case, the probability is equal to the neighbors in the same sector divided by the total number of neighbors. Thus, it only considers the neighbors of node v_i.

In the second one, the probability is equal to nodes in the same sector divided by all the nodes. Therefore, it considers all the nodes in the graph.

From the result, we can see that the second evaluation method has a higher value.

Question 5.

This question requires construction of correlation as well. However, the data needed here is different. We need to use all the Monday's data instead of the weekly data. The MST from the correlation graph is shown below:

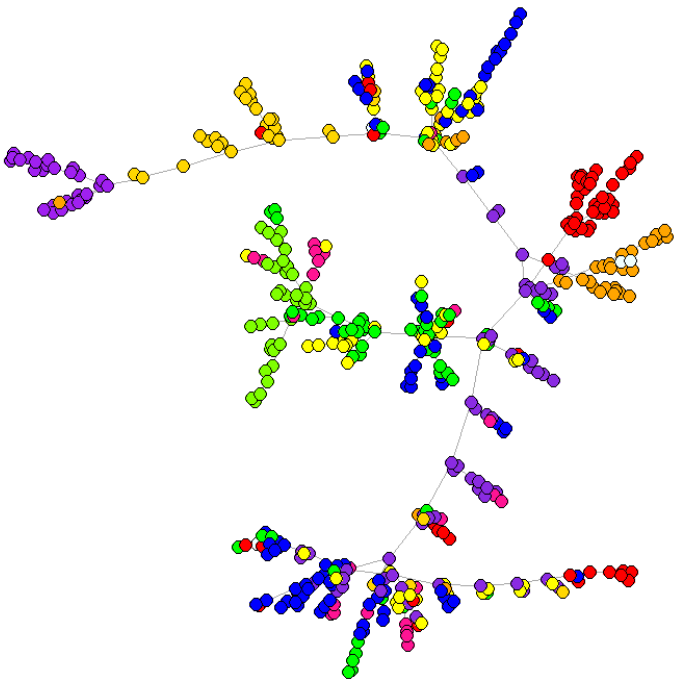


Figure 4. MST of the correlated graph based on weekly data

Similar to the former MST graph, nodes with the same color are closer and more likely to be connected, while nodes form different sectors are more likely to be separated.

However, in this MST graph for weekly data, this pattern is more obvious. The nodes with different colors are less likely to be connected with each other than the ones in the former

MST graph. So, the sectors' features are more obvious considering the connections between their nodes with regard to the weekly data. In other words, nodes with the same sectors are more likely to be clustered together in weekly data than in daily data.

2 Let's help Santa

2.2 Build Your Graph

Question 6.

In this part, we process the given data and build a graph based on the processed data. The steps are as below:

1) Extract data from san_francisco-censustracts-2017-4-All-MonthlyAggregate.csv (only December);
2) Merge duplicate edges by averaging their weights;
3) Extract information from san_francisco_censustracts.json. Add "display name" and "location" (mean of the coordinates of the polygon's corners) attributes to the vertices;
4) Extract and keep the giant connected component of the graph (as G).

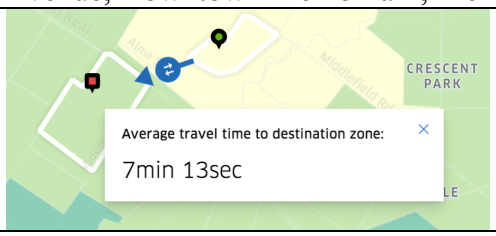
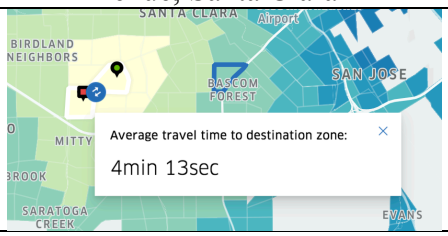
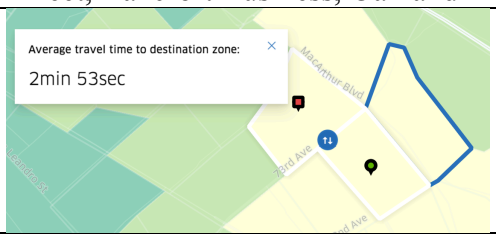
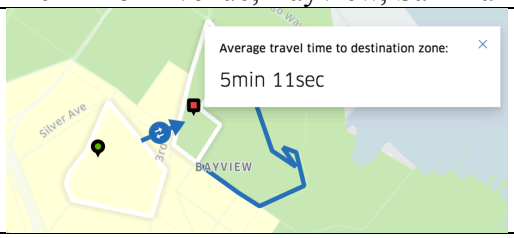
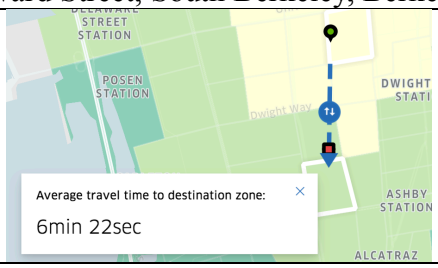
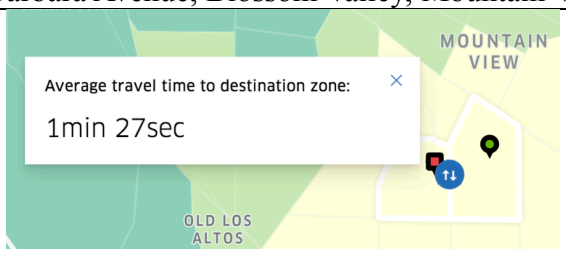
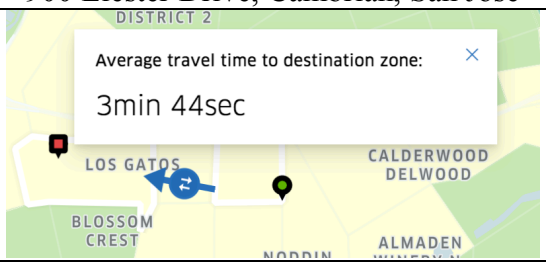
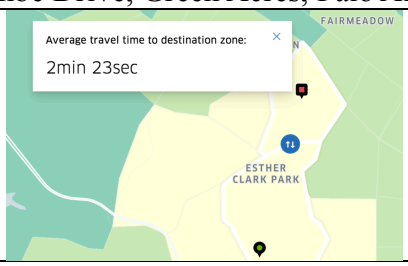
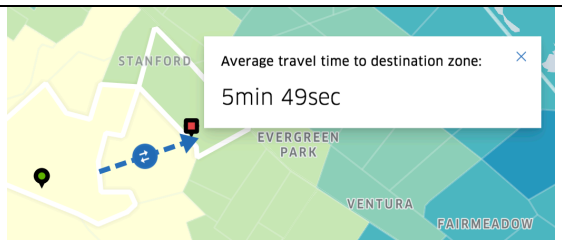
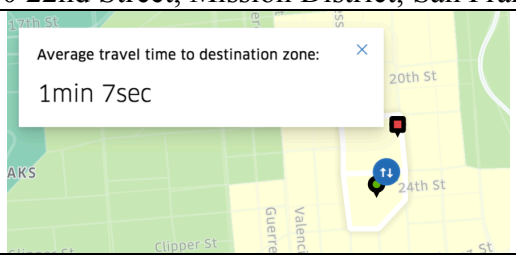
- The number of nodes in G: 1880
- The number of edges in G: 311802

2.3 Traveling Salesman Problem

Question 7.

In this part, we use the build-in function spanning_tree() to build a minimum spanning tree (MST) of graph G. Below are part of the edges with the street addresses of the two endpoints:

Table 1. part of edges in MST of graph G

Total: 1880 nodes, 1879 edges	
300 Sherwood Way, Linfield Oaks, Menlo Park -- 800 Live Oak Avenue, Downtown Menlo Park, Menlo Park	600 Hobart Terrace, Santa Clara -- 3400 Mauricia Avenue, Santa Clara
	
2600 79th Avenue, Eastmont, Oakland -- 2500 Church Street, Bancroft Business, Oakland	200 Topeka Avenue, Silver Terrace, San Francisco -- 1600 McKinnon Avenue, Bayview, San Francisco
	
2200 McGee Avenue, Central Berkeley, Berkeley -- 1600 Ward Street, South Berkeley, Berkeley	600 Hans Avenue, Cuesta Park, Mountain View -- 1000 Barbara Avenue, Blossom Valley, Mountain View
	
1700 Los Gatos Almaden Road, Cambrian, San Jose -- 4900 Elester Drive, Cambrian, San Jose	26200 Catharine Court, Los Altos Hills -- 4100 Coulombe Drive, Green Acres, Palo Alto
	
100 Campus Drive, Stanford -- 600 Campus Drive, Stanford	2800 Folsom Street, Mission District, San Francisco -- 2900 22nd Street, Mission District, San Francisco
	

From the table above, we find the results intuitive. On the one hand, we get the relationship between nodes and edges in MST: $E = V - 1$. MST is a kind of tree, so it should obey the property of a tree that the number of edges is one less than the number of nodes. Besides, there is no cycle in a tree. Therefore, the relationship is intuitive.

On the other hand, from the 10 figures above, we find that the two endpoints of an edge are very near, almost in adjacent polygon or in adjacent of adjacent polygon. Besides, the travel time of the two polygons are almost in yellow (0-5 mins) or light green (5-10 mins). It also verified the theory of MST that the tree can connect all nodes in the graph with the lowest total weight. Each round generating a new node, it will check all surrounding nodes of the existed tree and choose one node which is the nearest to the whole tree. So the edges tend to connect the adjacent zone, or if the adjacent zone already existing in the tree, choosing the next adjacent-level zone. In summary, the results are intuitive and consistent to the theory of MST.

Question 8.

In this part, we randomly sampled 1000 triangles in graph G and calculated the percentage. Below are the steps:

1) Use random.randint() function to pick up 3 integers (nodes) and sort the list;
2) If the current list of nodes has already been picked up, then we re-picked again until we find one new combination;
3) Count the edges among those three nodes. If the number of edges is less than three (some pair of nodes don't have direct path), then we re-picked again;
4) Check if current nodes satisfy triangle inequality or not: $a + b > c$ and $b + c > a$ and $a + c > b$;
5) Calculate the percentage of triangles that satisfy the triangle inequality.

Below is part of our results:

```
========================================experience 0========================================
edge 1: 1300 Tolteca Court, Weibel, Fremont -- 1900 Hughes Drive, Monument Corridor, Concord
weight: 2296.5

edge 2: 500 El Paseo Drive, Sobrante Park, Oakland -- 1900 Hughes Drive, Monument Corridor, Concord
weight: 2050.48

edge 3: 500 El Paseo Drive, Sobrante Park, Oakland -- 1300 Tolteca Court, Weibel, Fremont
weight: 1806.92

satisfy the triangle inequality
========================================experience 1========================================
edge 1: J. Arthur Younger Freeway, Foster City -- 2500 10th Avenue, Bella Vista, Oakland
weight: 2311.5150000000003

edge 2: J. Arthur Younger Freeway, Foster City -- 1800 Scott Street, Western Addition, San Francisco
weight: 2721.88

edge 3: 1800 Scott Street, Western Addition, San Francisco -- 2500 10th Avenue, Bella Vista, Oakland
weight: 2036.835

satisfy the triangle inequality
```

```
====================================experience 998====================================
edge 1: 100 Octavia Street, Western Addition, San Francisco -- 3700 Callan Boulevard, Westborough, South San Francisc
o
weight: 1512.1999999999998

edge 2: 100 Edgemar Street, Crocker, Daly City -- 100 Octavia Street, Western Addition, San Francisco
weight: 1298.68

edge 3: 100 Edgemar Street, Crocker, Daly City -- 3700 Callan Boulevard, Westborough, South San Francisco
weight: 1050.15

satisfy the triangle inequality

====================================experience 999====================================
edge 1: 3800 Elston Avenue, Glenview, Oakland -- 4200 Maybelle Avenue, Redwood Heights, Oakland
weight: 516.335

edge 2: 3800 Elston Avenue, Glenview, Oakland -- 1200 Paru Street, West Alameda, Alameda
weight: 1208.19

edge 3: 1200 Paru Street, West Alameda, Alameda -- 4200 Maybelle Avenue, Redwood Heights, Oakland
weight: 1271.835

satisfy the triangle inequality
================================================================================
satisfying rate: 0.943
```

Figure 5. part of test of triangle inequality

The percentage of triangles that satisfy the triangle inequality is about 94.3%.

Question 9.

In this part, we use 2-approximate algorithm to compute the approximate TSP cost. Below are the steps:

1) Find the minimum spanning tree T;
2) Create an Eulerian spanning graph by using two copies of each edge of T;
3) Find an Eulerian walk;
4) Find a 2-approximate tour;
5) Check each edge in the embedded tour. If it doesn't exist in the original graph G, then substitute with the shortest path (Dijkstra algorithm) between those two nodes.

The relationships of costs are as below:

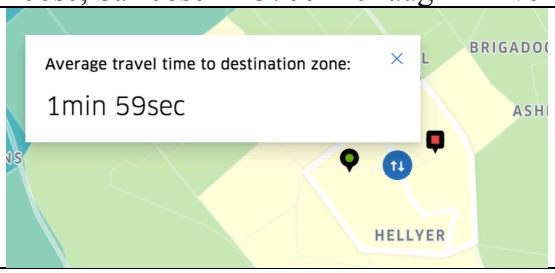
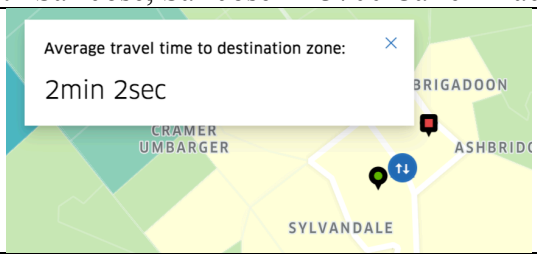
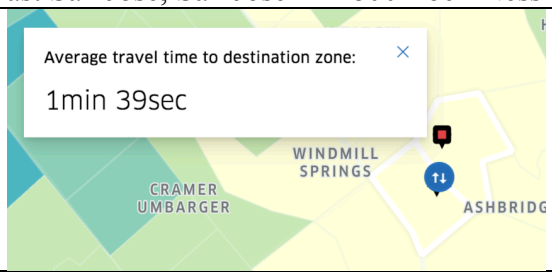
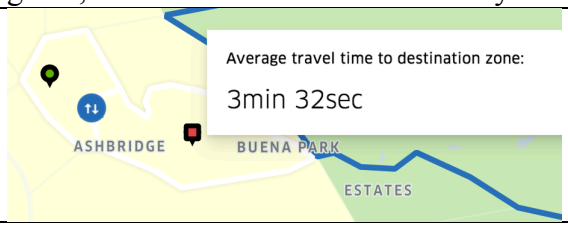
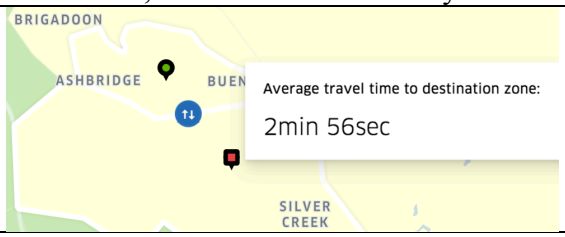
$$\text{MST Cost} < \text{Optimal TSP Cost} < \text{Approximate TSP Cost} < 2 \times \text{MST Cost}$$

Then we can derive the lower bound and upper bound:

$$1 < \rho = \frac{Approximate\ TSP\ Cost}{Optimal\ TSP\ Cost} < \frac{Approximate\ TSP\ Cost}{MST\ Cost} = 1.675$$

Part of the trajectory is as below:

Table 2. part of sequence of street addresses

	Path
1	3300 Brodie Drive, South San Jose, San Jose → 3700 McLaughlin Avenue, South San Jose, San Jose
	
2	3700 McLaughlin Avenue, South San Jose, San Jose → 3700 Carick Place Way, East San Jose, San Jose
	
3	3700 Carick Place Way, East San Jose, San Jose → 1800 Loch Ness Way, Evergreen, San Jose
	
4	1800 Loch Ness Way, Evergreen, San Jose → 2200 Terrena Valley Drive, East San Jose, San Jose
	
5	2200 Terrena Valley Drive, East San Jose, San Jose → 5300 Canyon Hills Lane, East San Jose, San Jose
	

From the above table, we find that the results of path are intuitive. The best strategy for Santa is to travel as more as nodes around each area. The two endpoints on the edges of path are all adjacent, so that Santa can travel all places on the map with the optimal cost (least sum of travel time).

Question 10.

The number of edges in the final trajectory is 1926. We use Tableau to visualize the trajectory that Santa has to travel.

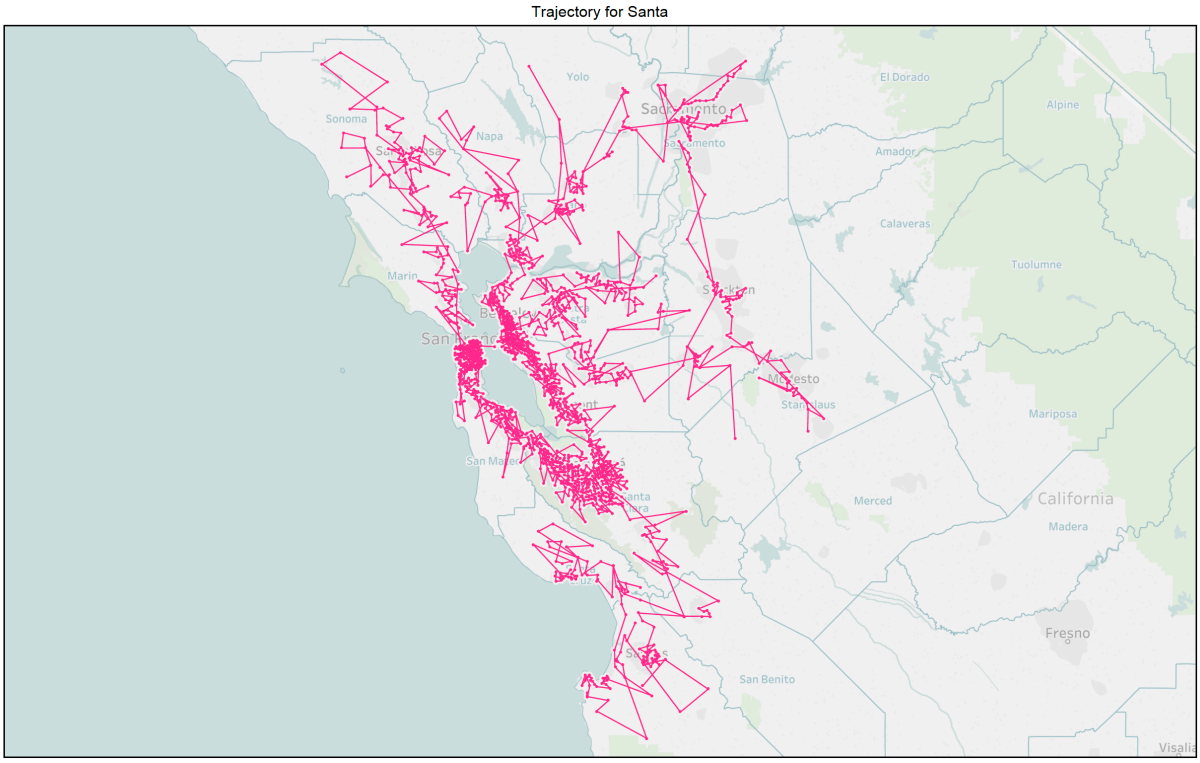


Figure 6. trajectory for Santa (full view)

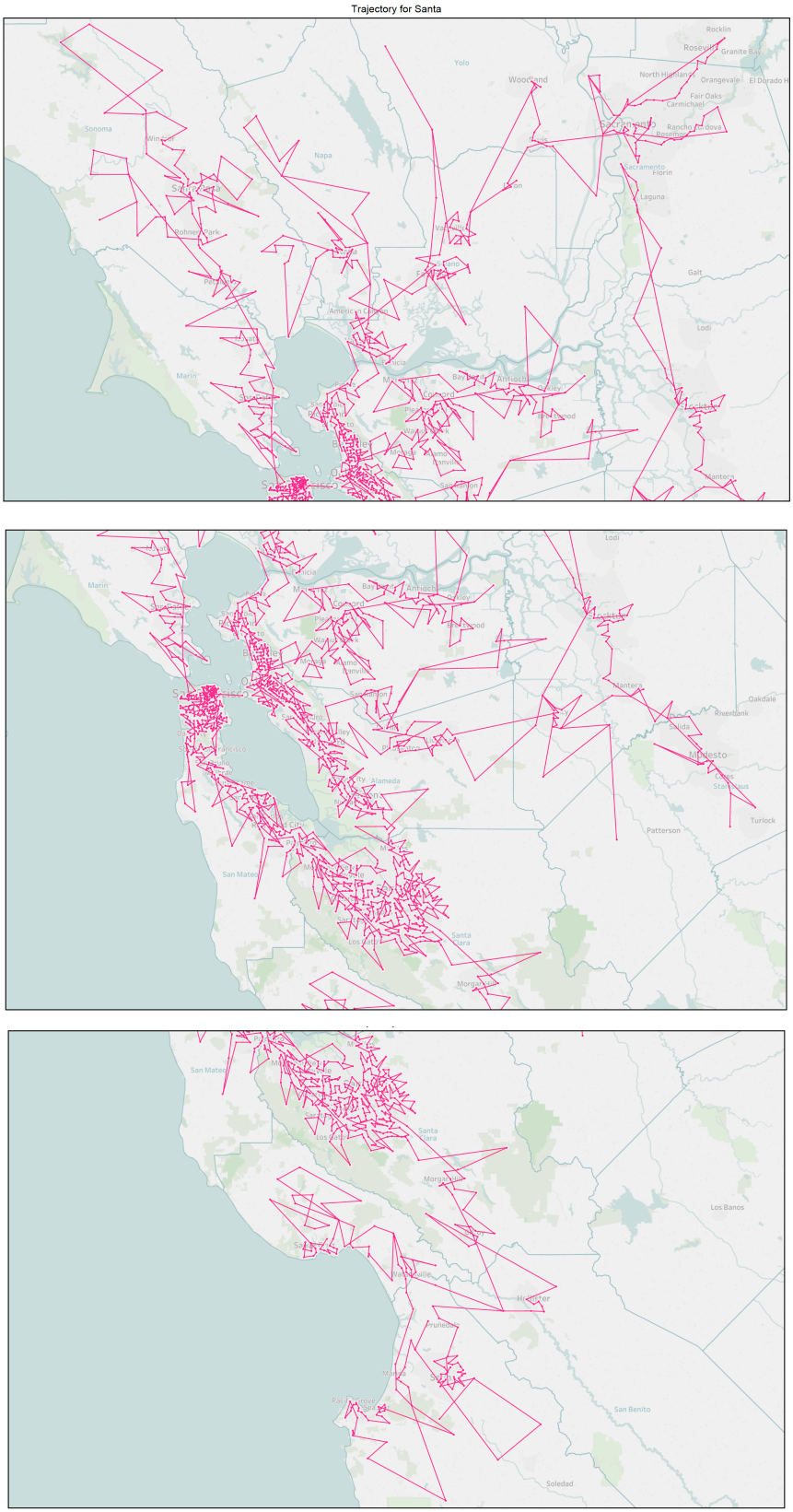


Figure 7. trajectory for Santa (parts view)

3 Analyzing the Traffic Flow

In December, there is a sport event between Stanford University and University of California, Santa Cruz (UCSC), and the event is in UCSC, so many fans will drive from Stanford to UCSC. In this part, we would like to analyze the traffic that can flow from Stanford to UCSC.

3.1 Estimate the Roads

We want to estimate the maps of roads without using the real road data. Therefore, we apply Delaunay triangulation algorithm on the node coordinate and use the edge to represent the roads.

Question 11

We first load the given coordinate file. For each nodeID, there is a boundary with multiple points with coordinates. We use the mean coordinate of the boundary to represent the node coordinates. Then, we take the GCC node from question 6, and search the coordinates for each node to create a coordinate array. Then, we apply Delaunay triangulation algorithm on those vertices to create the road mesh of San Francisco Bay Area, and the result is shown below.

In the figure below, we can observe the green spots, which are the nodes. The nodes are more in the city area and less in countryside, because for Uber dataset, if a place with more service/data, the place should be divided into more section for analyzing purpose. Additionally, the edges are generated by math, so there are many long distance edges, which cannot represent the real road, so we have to make further process to let the road mesh more real.

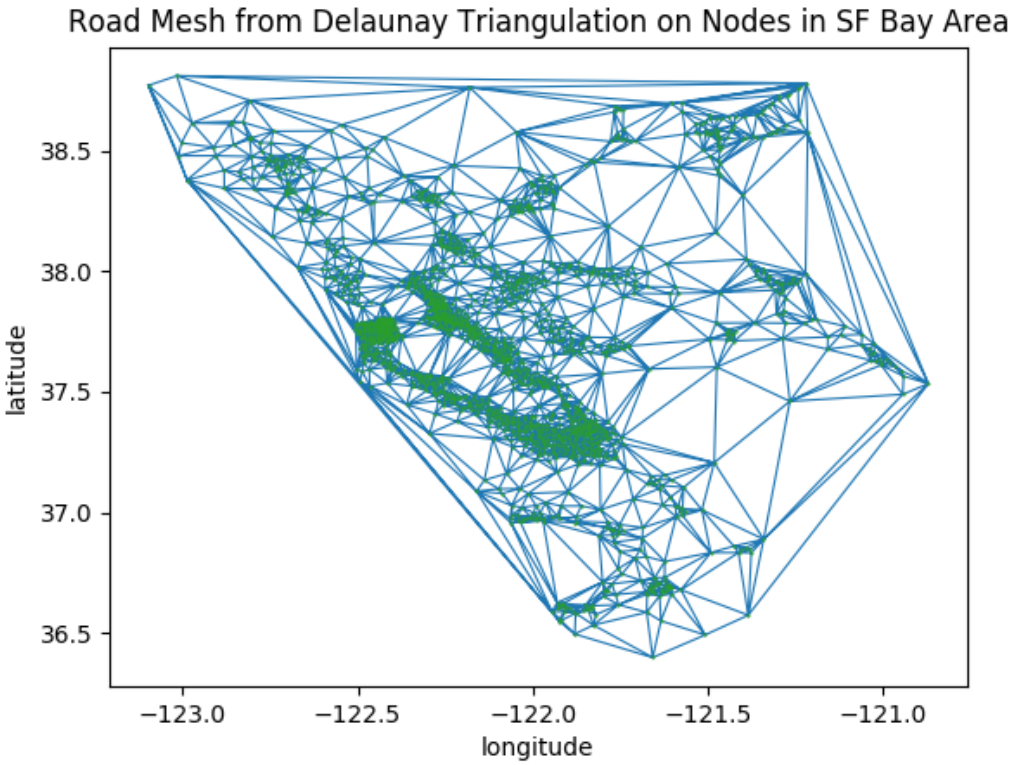


Figure 8. Road Mesh of SF Bay Area

Then we load the san_francisco-censustracts-2017-4-All-MonthlyAggregate.csv, and take only December as our reference (December Data). From the December Data, we can get the time of most edges (by triangulation) and calculate Euclidean distance of those edges. With the data, we create a table which contain source ID, destination ID, time, and distance. For the missing edges by triangulation, we load table as directed graph in r (edge weighted by time), and find time and distance of the missing edges by calculating the shortest path(time) and the corresponding path distance. Then, we get the complete table of those edges and create a graph G_Δ whose nodes are different locations and whose edges are produced by triangulation. The figure of G_Δ in actual coordinates are shown below.

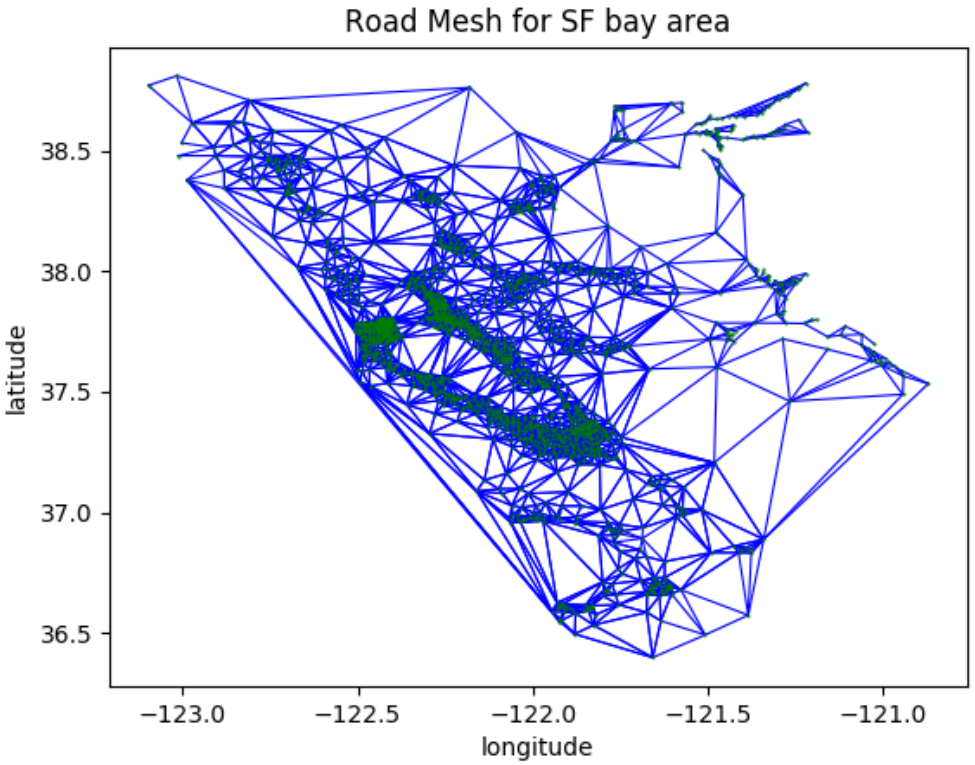


Figure 9. G_{Δ} in Actual Coordinates

3.2 Calculate Road Traffic Flows

Question 12

In this question, we want to sue simple math to calculate the traffic flow for each road in terms of car/hour.

We use the following assumptions.

- Each degree of latitude and longitude ≈ 69 miles
- Car length ≈ 5m = 0.003mile
- Cars maintain a safety distance of 2 seconds to the next car
- Each road has 2 lanes in each direction

Here is how we derive the formula to calculate traffic flow in terms of distance and time.

For a given road, given road distance and passing time as *d mile* and *t hour*.

Take car velocity as *v mile/hour*, $v = \frac{d}{t}$

The safety distance is 2 second of car speed, which is $\frac{v}{3600} * 2 = \frac{v}{1800} mile$

For the number of cars remain in a road is $\frac{d}{0.003 + \frac{v}{1800}}$ cars

Traffic flow = $\frac{d/t}{0.003 + \frac{v}{1800}} = \frac{v}{0.003 + \frac{v}{1800}} = \frac{1}{\frac{0.003}{v} + \frac{1}{1800}}$ cars/hour

Each road has two lanes, so the traffic flow of each road is $\frac{2}{\frac{0.003}{v} + \frac{1}{1800}}$ cars/hour

In our data, time and distance are given, by applying velocity into the formula, we can traffic flow for each road and use them on later question.

3.3 Calculate the Max Flow

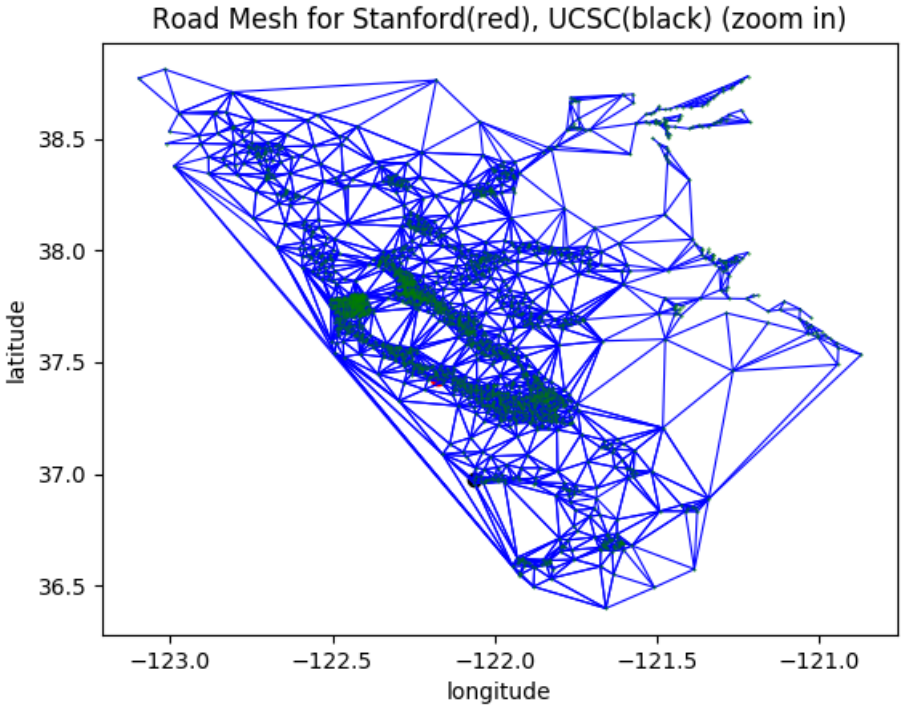
Question 13

In this question, we are asked to calculate the maximum number of cars that can commute per hour between the two points. Also calculate the number of edge-disjoint paths between the two points.

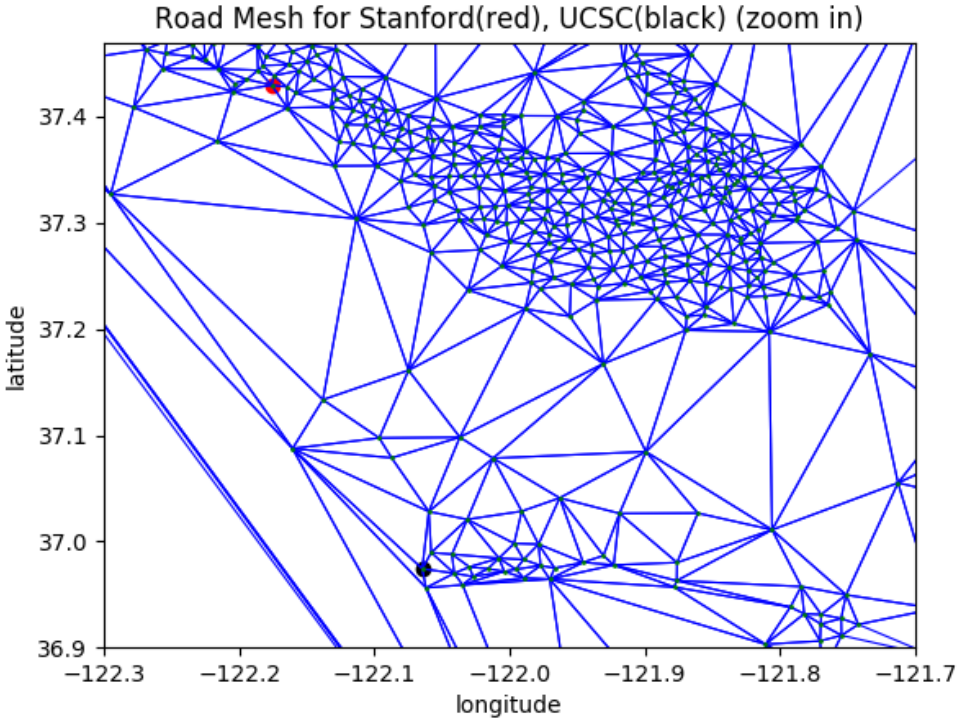
- Source address: 100 Campus Drive, Stanford
- Destination address: 700 Meder Street, Santa Cruz

The max car flow is constrained by the amount of cars flow out from Stanford, flow in to UCSC, and the intermediate path. By applying the R function max_flow to the graph made of source id, destination id and the car flow table (from question 12), we can get the maximum number of cars that can commute per hour from Stanford to UCSC is 14795.81 cars/hour.

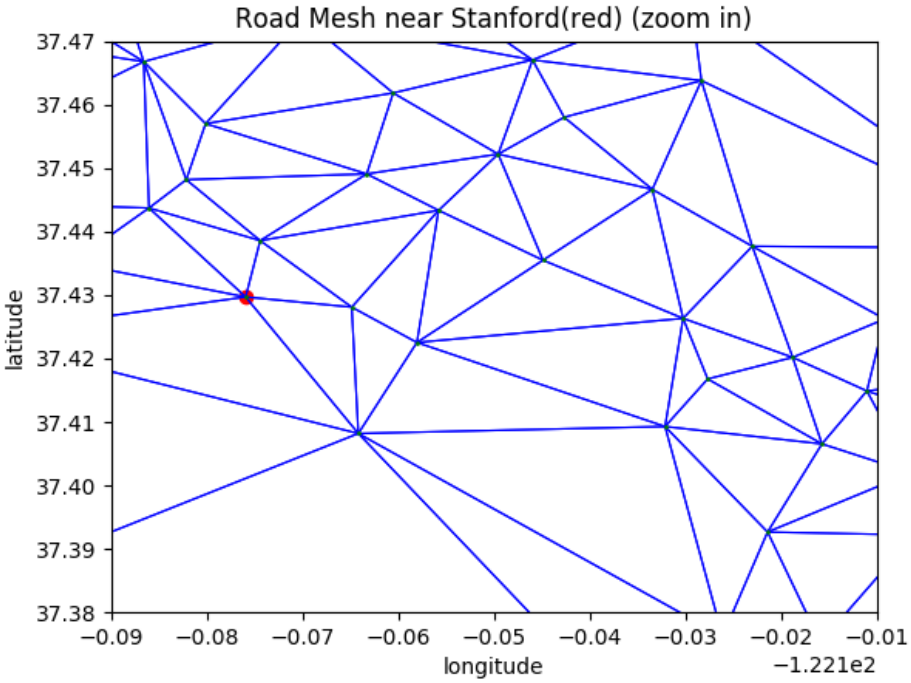
From the figure below (road maps), for Stanford and UCSC, there are 6 and 5 neighbor nodes. From Graph G_Δ below, the nodes number pointing out from Stanford (node ID = 2607) is 6 and the nodes number pointing into UCSC (node ID = 1968) is 5. Therefore, there are 5 edge-disjoint paths because from the origin(Stanford), people have to choose from 6 roads, and to the destination(UCSC), people have to choose 5 roads. The 6 starting road cannot combine with the 5 ending road to form edge-disjoint paths, so we have to choose the minimum of the two number, which is 5, as the number of edge-disjoint paths, which is same as we see on road map.



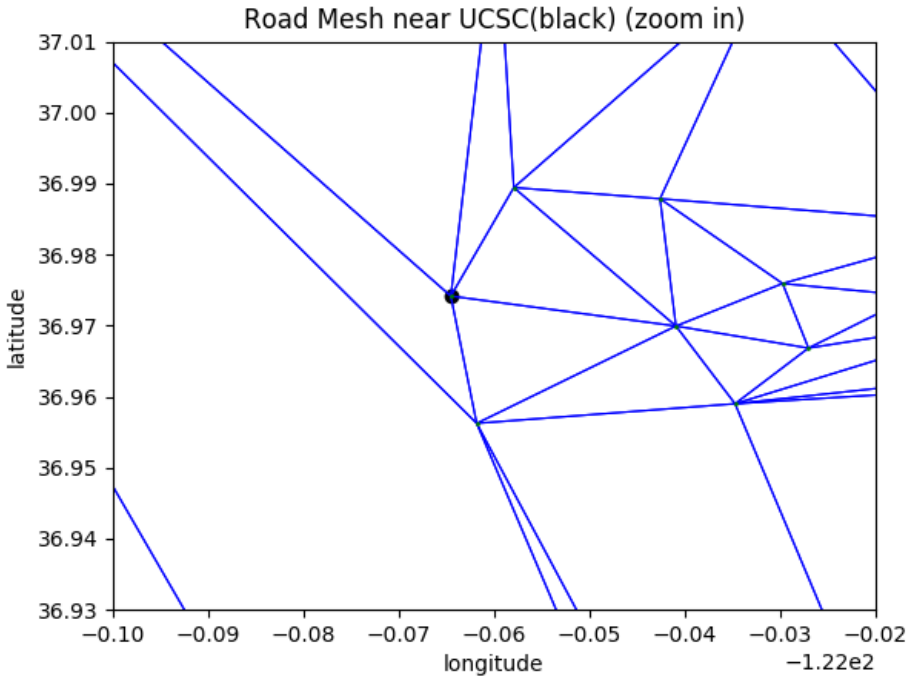
(a) Normal Scale of the Map



(b) Map Focused on Stanford and UCSC

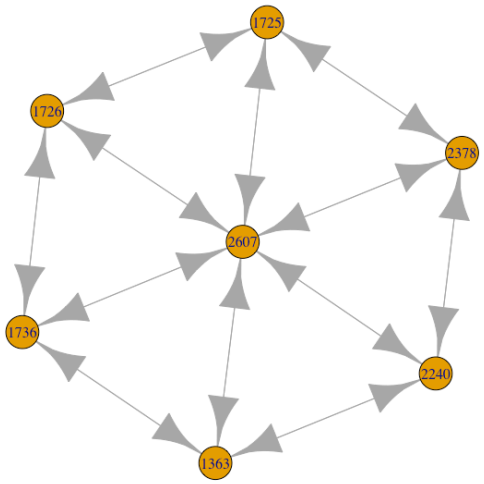


(c) Map Focused on Stanford

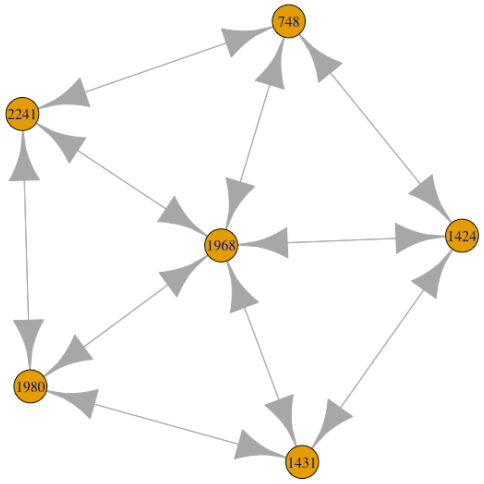


(d) Map Focused on UCSC

Figure 10. Graph G_Δ on Actual Coordinates



(a) Stanford (#2607) and Neighbor Nodes



(b) UCSC (#1968) and Neighbor Nodes

Figure 11. Graph G_{Δ} on Stanford, UCSC and Their Neighbor Nodes

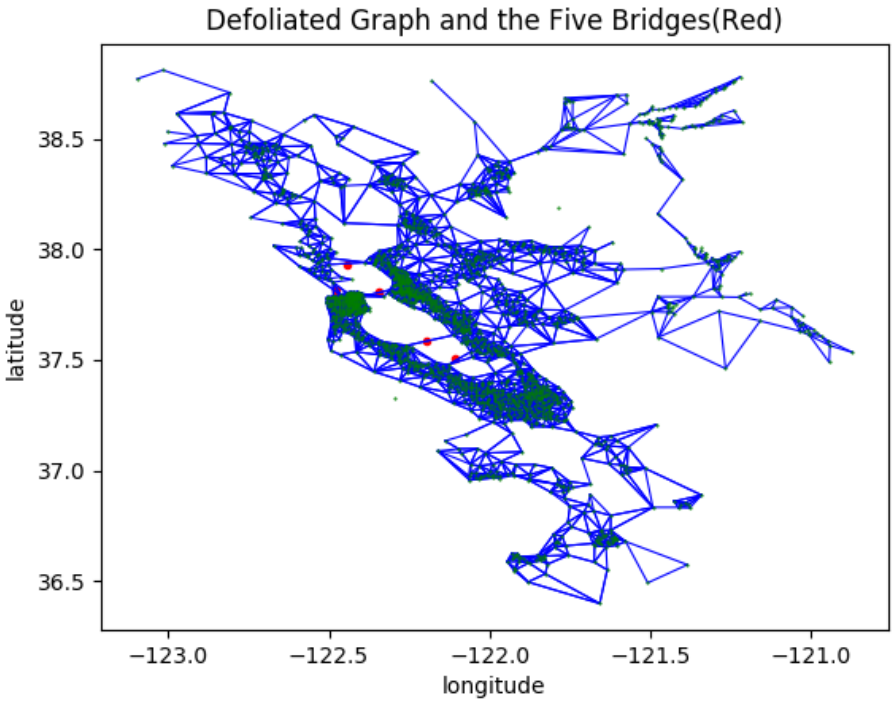
3.4 Defoliate Your Graph

Question 14

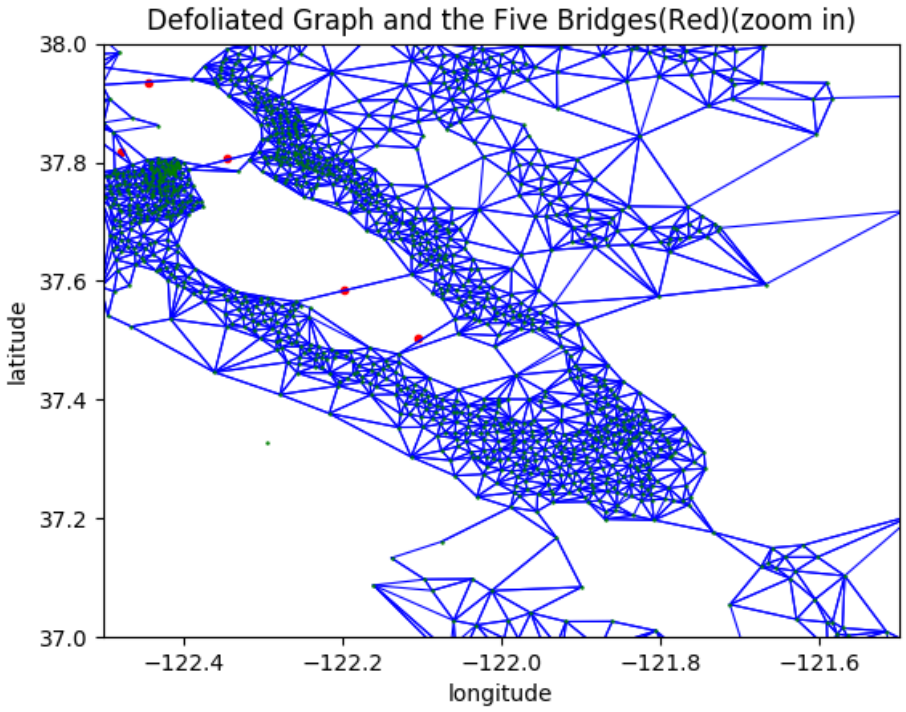
In this question, we are asked to remove the fake edges for our G_Δ by applying time threshold. Because for long time/distance travel, the edge cannot be straight line, the path should contain many short edges, so I trim the edges with traveling time above the time threshold. By applying time threshold with low value, we can trim many edges including real edges, so we refer to the 5 bridges provided to carefully select our time threshold. Because to travel between the end of two edges, the shortest distance/time should be by bridge, so bridge is an appropriate method to judge the edge real or fake. Here are the five bridges.

- Golden Gate Bridge: [[-122.475, 37.806], [-122.479, 37.83]]
- Richmond, San Rafael Bridge: [[-122.501, 37.956], [-122.387, 37.93]]
- San Mateo Bridge: [[-122.273, 37.563], [-122.122, 37.627]]
- Dambarton Bridge: [[-122.142, 37.486], [-122.067, 37.54]]
- San Francisco - Oakland Bay Bridge: [[-122.388, 37.788], [-122.302, 37.825]]

By searching the five bridges provided, the top 2 longest time to pass a bridge are Dambarton Bridge (791 and 530 second) and San Francisco - Oakland Bay Bridge(619 and 493 second). In the beginning, I choose 850 second as threshold and find that there are still many fake bridges, so I reduce the threshold to 750 second to create $\tilde{G}_\Delta$, this would remove one direction of Dambarton Bridge but the bridge edge still remains. From the figure below, compared to G_Δ in actual coordinates, there are many long distance edges removed. For long distance travel, they should follow many short paths to complete the travel. For example, if A to B takes 2000 second, there should be some spot to pass by, like A to C, C to D, and D to B. Additionally, from the figure below, we can observe that the five bridges provided are preserved, and many fake bridges are removed.



(a) Normal Scale of Maps



(b) Map Focused on Bay Area

Figure 12. Defoliated Graph $\tilde{G}_{\Delta}$ on Actual Coordinates and the Five Bridges

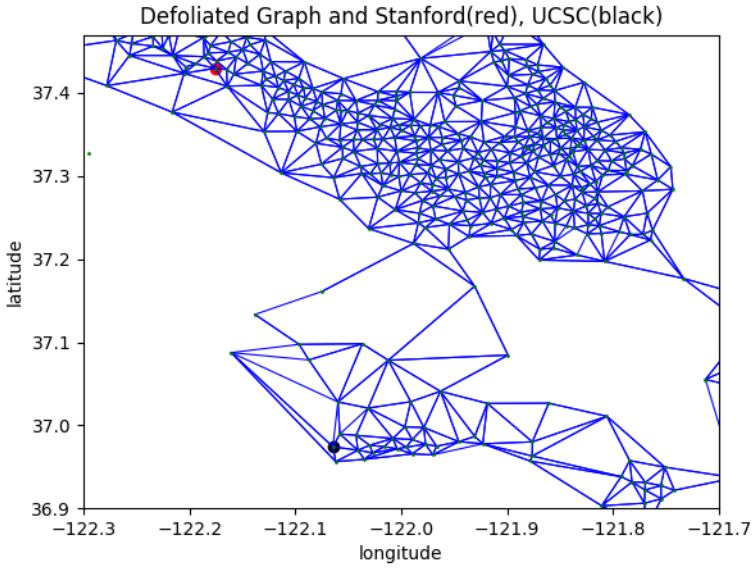
Question 15

From the Defoliated Graph $\tilde{G}_{\Delta}$ we generated from question 14, we can get the maximum number of cars that can commute per hour (car flow) from Stanford (node ID = 2607) to UCSC (node ID = 1968) is 12062.42 cars/hour. Compared to G_{Δ}, the maximum car flow is reduced by 2733.39 cars/hour. This result is because we remove some fake edges and this influence the car flow of the paths. In our graph, from Stanford to UCSC, there are many intermediate edges, this may not affect the max car flow, but the neighbor nodes of Stanford and UCSC have lots of effect on it, because one neighbor edge is removed by time threshold, which is edge (nodeID 1431 to 1968).

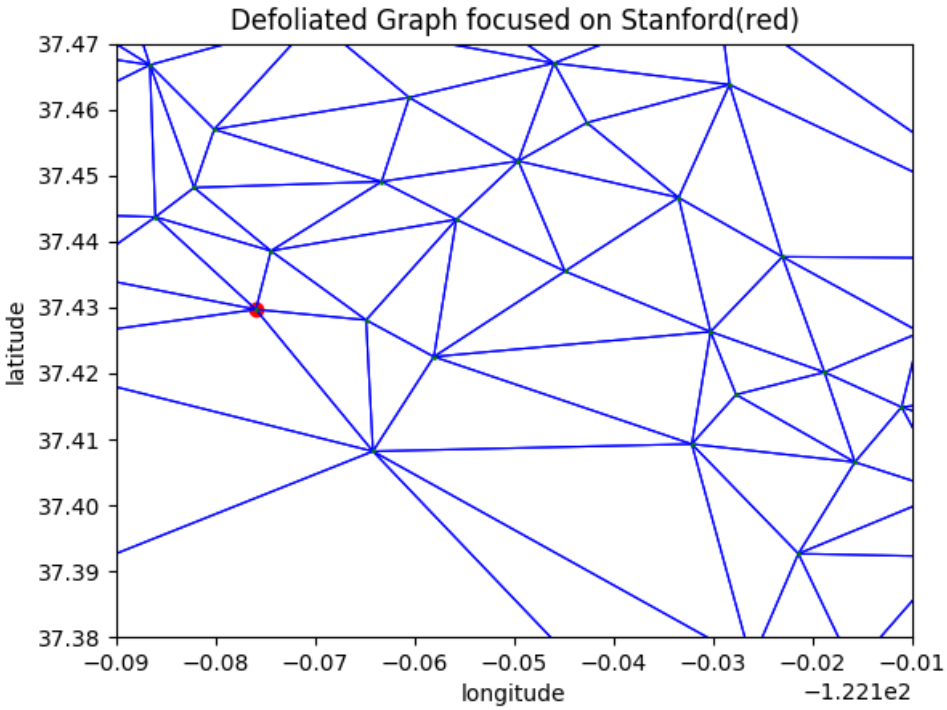
From the figure below (road maps), for Stanford and UCSC, there are 6 and 5 neighbor nodes same as the results in question 13, so we assume there are 5 edge-disjoint paths from Stanford to UCSC, but actually it is not 5. This is because in the road maps, if there exists an edge between two nodes, no matter flows in or out, but actually for one edge flows into UCSC is removed as the graph shown below.

From Graph $\tilde{G}_{\Delta}$ below, the nodes number pointing out from Stanford (node ID = 2607) is 6 and the nodes number pointing into UCSC (node ID = 1968) is 4. For UCSC, originally in G_{Δ}, there are 5 node pointing into it, but in $\tilde{G}_{\Delta}$, the edge (nodeID 1431 to 1968) is removed by the time threshold. However, The edge (nodeID 1968 to 1931) is still valid in G_{Δ}, so it still shows an edge in $\tilde{G}_{\Delta}$ on actual coordinates.

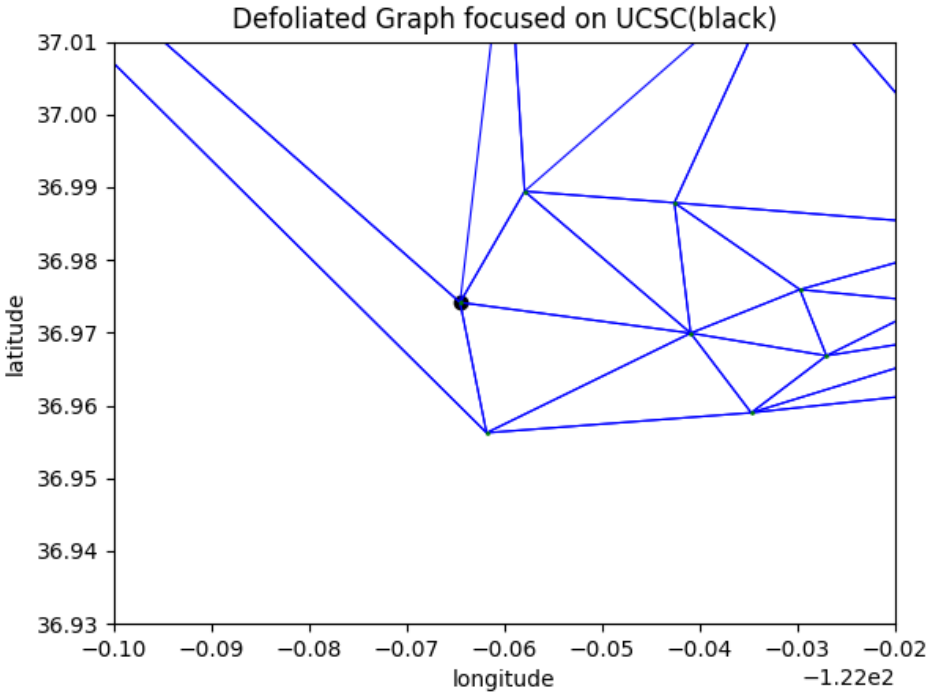
Because in $\tilde{G}_{\Delta}$, there are 6 neighbor nodes pointing out from Stanford and 4 neighbor nodes pointing into UCSC, so there are 4 edge-disjoint paths from Stanford to UCSC.



(a) Map focused on Stanford and UCSC

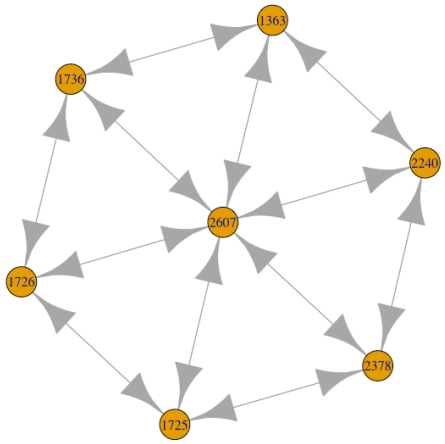


(b) Map focused on Stanford

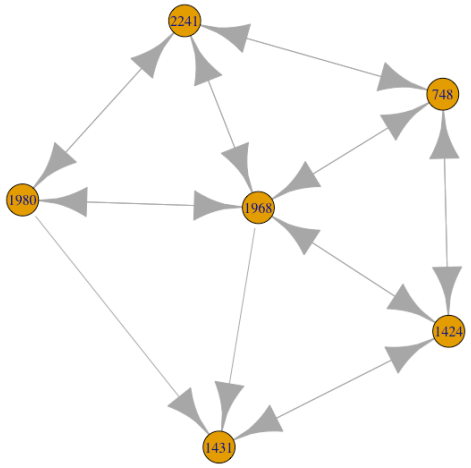


(c) Map focused on UCSC

Figure 13. Defoliated Graph $\tilde{G}_{\Delta}$ on Actual Coordinates



(a) Stanford (#2607) and Neighbor Nodes



(b) UCSC (#1968) and Neighbor Nodes

Figure 14. Graph $\tilde{G}_{\Delta}$ on Stanford, UCSC and Their Neighbor Nodes