NYC Optimal Transport and Ridesharing

by

Jiahui Ji

An honors thesis submitted in partial fulfillment
of the requirements for the degree of
Bachelor of Science
(Honors Statistics)
at the University of Michigan
2018

Supervisor: Dr. Long Nguyen

GSI: Aritra Guha

A C K N O W L E D G M E N T S

I want to gratefully acknowledge Professor Long Nguyen, Graduate Student Instructor Aritra Guha for providing instructions, resources, and assistance on this project. I would also like to acknowledge my partner Zui Chen for her input on the work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

NYC Optimal Transport and Ridesharing

by

Jiahui Ji

Supervisor: Dr. Long Nguyen

The motivation of this paper is the hope to enhance travel efficiency, reduce energy consumption and greenhouse gas emission. We first focus on the optimal transport problem and approaches. Sinkhorn's algorithm is applied with two separate assumptions of locations, discrete and semi-discrete. The next chapter focuses on the problem of ridesharing. Trip duration time is predicted by some regression models. It also introduces a dynamic ridesharing problem, which pulls certain trip-satisfying criterion together. Then a statistical model of shared rides and individual trips is described, with parameter estimations made from the frequentest approach. The last chapter concludes with the exploration of the above two problems, lists some limitations in the model, and provides some future work suggestions.

CHAPTER 1

Introduction

1.1 Motivation

New York is one of the most populous cities in the world. In order to serve the transportation need of 8.5 million people, complex public transportation systems are built. Taxi, serving as an easy and convenient transportation mode, is an important supplement of public transportation. On record, there are around 12 million trip records for yellow taxi and 1.5 million for the green taxi each month, which means on average there are approximately 45,000 taxi users daily in the city. An analysis of large quantities of trip record data could help learn the taxi demand distribution, big trends of traffic flow, traveling behaviors of passengers, cost and profit of taxi companies, etc. Further, the result could potentially assist with decision-making for taxi companies and the government to provide better public transportation.

With such large amount of taxi trips, two aspects of concerns are brought to my attention. One is the travel efficiency of taxis. Since the overall demand from passengers is greater than the available taxis and drivers, each taxi must travel around in the city a lot to pick up and drop off passengers distributed everywhere. Even though green taxis only pick up passengers in certain boroughs, the total active area is still quite large. When the last drop-off location is far from the next pick-up location, the taxi has to travel vacantly for a relatively long time, which is a waste of time and resources. It would be ideal to connect the trips which one's starting time and location are close to the other one's ending time and location, so that one taxi could serve both of them back to back. It is also a common phenomenon in cities that in some areas taxis fall in short with a high volume of requests, but in other areas, taxis are waiting for requests or running vacantly. A good strategy is needed to distribute taxis in geographical and time-space appropriately, according to the demand level.

Another aspect is the energy consumption and environmental impact of motor vehicles.

From the Energy Consumption Estimates table on the U.S. Energy Information Administration website [12], New York is ranked #5 among all the states on petroleum consumption in 2015. More than three-fourths of petroleum products consumed in New York are used in the transportation sector. Accompanied by the great amount of petroleum use is the greenhouse gas (i.e.CO2) emission. In figure 1.1, fuel combustion from transportation alone contributes to over 40% of the total CO2 emission from all the sectors. And those CO2 accounts for 83% of total greenhouse gases. From figure 1.2, the proportion of greenhouse gas emission from transportation in the entire U.S. is only 28%, but the percentage in New York is 34%, suggesting the emission from transportation in New York is more intensive than in most other states.

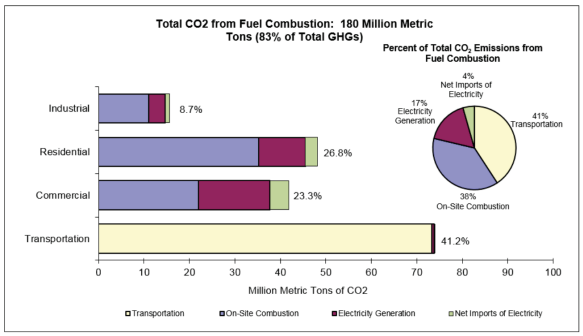

Figure 1.1: 2014 CO2 Emissions from Fuel Combustion by End Use Sector (Includes Net Imports of Electricity)[8]

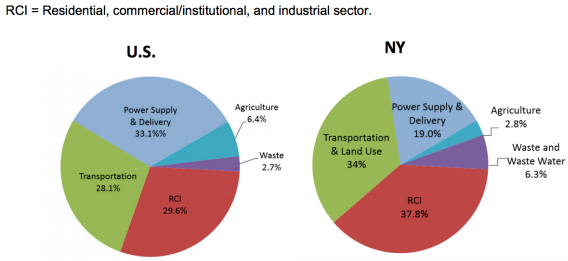

Figure 1.2: 2014 GHG Emissions by Sector: New York State and U.S.[8]

The statistics and figures reflect how much energy is consumed and greenhouse gas is generated by transportation in New York, in comparison with other states. There is no specific information on what proportion of consumption or emission comes from taxis. However, we realize that the impact of NYC motor vehicles on the environment is serious, and we hope to reduce it by decreasing unnecessary travelings in the city. It might be difficult to do with public transportation like subway, buses, since the travel plan is fixed. It is more possible to realize with taxis, because there is high flexibility in the travel plan.

1.2 Problems and Approaches

Motivated by the two concerns discussed above, I mainly addressed *Optimal Transport* and *Ridesharing* problem in this paper. Both of the problems aim at providing a better transportation plan than the current trip records. The proposed transportation plan shall have higher efficiency in riding passengers, less waiting time on both sides (passenger and driver), less vacant traveling and operational cost.

Chapter 2 mainly talks about *Optimal Transport*. It is the study that deals with the allocation of resources [9]. Considering the available taxis as resources, we are trying to allocate them to specific geological areas to meet passenger demands. With some techniques associated with the optimal transport theory, it is possible to compute the desired transportation plan.

Assume the drop-off and pick-up locations are spaces with some density distribution. I want to find the mapping from drop-off to pick-up locations that achieves minimum total cost. The algorithm for computing total cost is adapted from the paper "Sinkhorn Distances: Lightspeed Computation of Optimal Transportation Distances", where an entropy term added to the minimization. Two separate assumptions are made: one assumes both the drop-off and pick-up locations are discrete, and the other deals with the case where drop-off location is considered as a continuous variable, while the pickup location remains discrete. For different assumptions, I use a different method to work with the location data obtained from data sets. Then I visualize the transportation plan resulted from each approach.

Chapter 3 is about the *Ridesharing* problem. Taxi ride-sharing is an environmental-friendly and economic way of traveling. From the society's perspective, more taxi rides shared means that there are smaller traffic amount on the road, especially during rush hours. It helps to ease the problem of traffic congestion, particularly in big cities. Shorter total traveling distance brings decreased gasoline consumption and greenhouse gas emissions, which is beneficial for saving energy and improving the quality of the environment. From individual's point of view, typically sharing a ride with someone does not cause too much additional travel time, but could save a big portion of their taxi fare. Less traffic on the road also helps them to get to their destination faster. During busy time or at busy areas, even getting a taxi to ride is difficult. Ridesharing allows people to get taxis much easier.

We see that ridesharing is a promising approach. However, in reality, carpooling with someone is far less convenient than hailing a taxi directly on street. There are so many ongoing rides happening every minute from all over the place. Finding an optimal one going in the similar direction to join could be quite challenging. In the chapter, a simple ride searching criterion is proposed to pair up distinct trips that could be shared in practice. Then

from a modeling perspective, a statistical model is described to fit the trip data and simulate potential shared trips information.

In both of the problems, I utilized some statistical tools, such as kmeans, kernel density estimation, and many statistical distributions (multivariate Gaussian, Poisson, exponential, Dirichlet, etc). The R packages are of great use to the application of those techniques.

CHAPTER 2

Optimal Transport

2.1 Data Overview

The data of interest comes from the NYC Taxi & Limousine Commission website.[7] The data set includes information about three types of trips in New York City: yellow, green and FHV. The yellow taxi provides transportation for passengers in all five boroughs via street hails. Green taxi also called "Boro taxi" is affiliated with the Boro Taxi program. It can be dispatched to pick up passengers in northern Manhattan, the Bronx, Queens, Brooklyn and Staten Island and at the airports, but may not be available for street hails except Manhattan. All data is provided in csv format, each sheet containing the trip information for one month between 2009 and 2017.

We are mainly interested in yellow and green taxi trips, both of which have almost the same variables about trip time, location, fare, etc. However, the yellow trip data sets are much larger than the green data sets with more than 12 million trip records for each month, while the green data sheet only has around 1.5 million records. It is much easier to work with green trip data.

Before the middle of 2016, all the drop-off and pick-up locations are recorded as the exact longitude and latitude. Since July 2016, the drop-off and pick-up locations are recorded as `locationID`, which refers to specific Borough, zone, and service zone information. To work directly with the location points, it is preferred to have the longitude and latitude values. 2015 is the most recent year that all data sheets recorded location using longitude and latitude. Therefore, the green trip data of 12 months in the year 2015 are selected to perform the analysis.

There are 21 variables in each data set: VendorID (indicator of record provider), pickup datetime, dropoff datetime, Passenger count, Trip distance, pickup longitude, pickup latitude, RateCodeID (standard rate or other), Store and forward flag (if the trip record was held in vehicle memory before sending to vendor), dropoff longitude, dropoff latitude, payment

type (credit or cash or other), fare amount (time and distance fare), Extra (Miscellaneous extras and surcharges), MTA tax, improvement surcharge, tip amount, tolls amount, total amount (charged to passengers), and trip type (street hail or dispatch). The variables most relevant to this study are the date time, longitude and latitude for drop-off and pick-up.

The data visualizing figures only show the result of working with January 2015 green trip data. It is found that there is not much variation in the distribution of variables between any two months, and computing for all the data sets is quite slow.

We are firstly interested learning about the geological distribution of drop-off and pick-up locations on the real map of New York City.

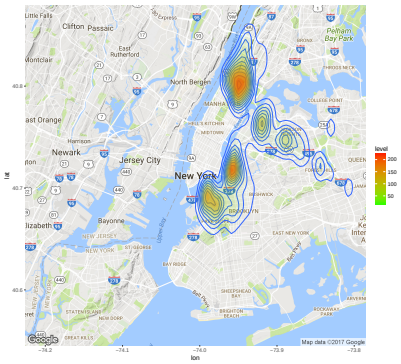

Figure 2.1: pick-up Density

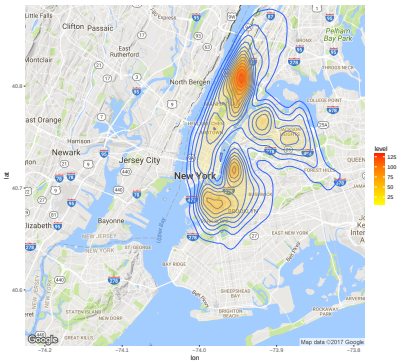

Figure 2.2: drop-off Density

In both figures, the scale of location density is shown as the gradient color on the map. Areas covered by more red color indicate a larger number of trips have drop-off or pick-up locations around here. In the pick-up figure, the dark red area is in northern Manhattan, a lighter red area is in northern Brooklyn, western Brooklyn and Queens also show some pick-up location distribution. The drop-off figure is similar, except that the trip locations are more intensely distributed in the Manhattan area, and the rest are distributed more sparsely and uniformly in other areas than in the pick-up figure.

The figure below gives a clearer look at the distinct drop-off and pick-up locations in terms of longitude & latitude in one plot, marked by red and green color respectively.

We can see that the red points are centralized in three irregular shapes of clusters in the north, middle part, and southwest. The green points are distributed more loosely and sparsely, but in Manhattan area, specifically Hell's Kitchen Midtown, it shows clear street patterns. The result corresponds to what we got in Figure 1. We may conclude that green taxi picks up from several neighboring boroughs and drops off mainly in Manhattan.

The data sets include intensive taxi trips, which means there is a great demand for taxis in New York City. A key question to be asked is how to find an efficient way to distribute taxis between trips so that the traveling and operational cost would be minimized, while all

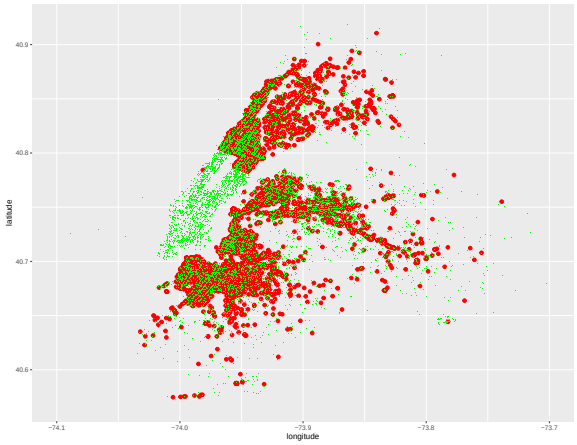

Figure 2.3: Location Points Distribution. The red points represent pick-up locations and green points represent drop-off locations.

the passengers get the requested rides as soon as possible. I would like to find an optimal transport plan that provides directions for vehicles from where they drop-off passengers to where they need to pick-up the next. Optimal means the total distance that vehicles travel is minimized.

Here is the outline for the rest of this chapter: In Background section 2.2 the problem is set up in the mathematical context and notations are introduced; the Algorithm section 2.3 gives a detailed explanation of the Sinkhorn-Knopp's fixed-point iteration algorithm; Sections 2.4 and 2.5 Discrete Approach and Semi-Discrete Approach talk about the procedure of performing each approach and their outcomes.

2.2 Background

2.2.1 Data Pre-Processing

The data obtained are individual trip records. The variable "dropoff _ datetime" specifies the exact time it was dropped off, same for the pickup_datetime. To study the transition between hours regardless of day and month, we parsed the trips with the same drop-off and same pick-up hour together. Each dataset contains the trip that was dropped off or picked up within the same hour. Therefore we obtained 24 data sets for drop-off and 24 data sets for pick-up.

2.2.2 Transportation Theory

It is used in mathematics and economics to study the optimal transportation and allocation of resources.[9] The problem was formalized by the French mathematician Gaspard Monge in 1781. In the 1920s, A.N. Tolstoi was one of the first to study the transportation problem mathematically. Major advances were made in the field during World War II by mathematician and economist Leonid Kantorovich.Consequently, the problem as it is stated is sometimes known as the Monge–Kantorovich transportation problem. Since our problem also deals with the allocation of resources and trying to find an optimal solution, we decided to work on this approach.

Monge and Kantorovich Problem Let X and Y be two metric spaces, c be a measurable function on X and Y, μ, ν be the probability density on X and Y, aim to find a map T: $X \to Y$ that the following function is minimized [9]:

$$\int_X c(x, T(x)) d\mu(x) \tag{2.1}$$

$T(x)$ is constrained by $T(\mu) = \nu$. Here T is the mapping that transfers some density in X to Y. Since we want the total X to be transferred to exactly Y, the constraint is set. $c(x, T(x)$ evaluates the transportation cost between x and $T(x)$ where x is one unit of mapping. To get the total cost, take the integral on the entire X.

2.2.3 Notations

- Mass Distribution r, c
  Assume drop-off and pick-up locations are two continuous space with various location points. At each point, we could count how many vehicles are at or around this location. The proportion of vehicles over the total vehicles at the same time is the *density* or *mass* for this location point. Hence obtain a *mass distribution* for drop-off location called r and one for pick-up location called c. r and c are vectors with the same length as the number of locations involved in each distribution. Note that since each component of r and c is a proportion, the sum of each of them is 1.

- Cost Matrix M
  In the transportation problem, cost refers to the travel distance. The cost matrix M is an $m \times n$ matrix, where m is the length of r and n is the length of c. The rows of M indicate drop-off locations and columns indicate pick-up ones. Each entry of M is

the pair-wise distance from a drop-off location to a pick-up location. e.g. M_{ij} is the distance between location i and location j.

- Mass Transfer Matrix P
  P is also an $m \times n$ matrix denoting the mass transfer from drop-off to pick-up locations. The rows of P indicate drop-off locations and columns indicate pick-up ones. P_{ij} is the mass that is transferred from location i to location j. For each drop-off location, the mass is the sum of mass transferred to all pick-up locations. Therefore the row sums of P should equal to the drop-off mass distribution vector r. For each pick-up location, the mass comes from the mass of all the drop-off locations, therefore the column sums of P equal to the pick-up mass distribution c. Since both the sum of r and sum of c is 1, can see that sum of P is also 1.

- Entropy Constraint h(P)
  This is a term not directly related to the problem. However, computing Sinkhorn's distance requires the term to be part of minimization. Entropy is a measure of randomness in the variable. The larger h(P) is, the more random or diffuse the mass transfer plan would be. The smaller h(P), the more set or directed the plan is. More mathematical interpretation of entropy is discussed in the Algorithm section.

2.2.4 Problem Setup

We want to find the optimal transportation plan between r and c given some fixed cost M. Now that we have the mass transfer matrix P, the cost matrix M for a unit of the mass transfer, so the total cost will be the dot product between P and M. The total cost:

$$\langle P, M \rangle = \sum_{i,j} p_{ij} m_{ij} \tag{2.2}$$

Define $U(r, c) = \left\{ P \in \mathbb{R}^{m \times n} | P\mathbf{1}_m = r, P^T \mathbf{1}_n = c \right\}$ $U(r, c)$ is the set of all the P that satisfy the row sums of P match r, column sums of P match c. P should only map from mass distribution r to c. The optimal transportation plan means the minimum cost, so our desired total cost is defined [2]:

$$d_M(r, c) = min_{P \in U(r,c)} \langle P, M \rangle \tag{2.3}$$

The desired P achieves above. The modification of the setup is done in the next section.

2.3 Algorithm

2.3.1 Entropy Regularization

2.3.1.1 Entropy and Kullback-Leibler divergence

Entropy is a measure of randomness in the variable. When entropy is maximized, the change of variable becomes uniform. When it is minimized, the variable has a set way of changing or moving.

For a vector r $\in \sum_m$, its entropy $h(r) = -\sum_{i=1}^{m} r_i \log r_i$

For a matrix $P \in U(r, c)$, its entropy $h(P) = -\sum_{i=1}^{m}\sum_{j=1}^{n} p_{i,j} \log p_{ij}$

Kullback-Leibler divergence (called KL) is defined as follows [2] :

Let P, Q $\in U(r, c)$, then $\mathbf{KL}(P||Q) = \sum_{ij} p_{ij} \log \frac{p_{ij}}{q_{ij}}$

An intuitive way to think of KL is how much randomness P has diverged from Q. In our problem, $Q = rc^T$ which indicates the plan where each trip is independent of one another, or the mass transfer is completely random. $\mathbf{KL}(P||Q)$ is the reduced randomness of P comparing to the maximized randomness.

To draw connections between $\mathbf{KL}(P||rc^T)$ and the individual entropy of P, r and c, we could actually show

$$\mathbf{KL}(P||rc^T) = -h(P) + h(r) + h(c) \tag{2.4}$$

Proof:

$$\begin{aligned}
\mathbf{KL}(P||rc^T) &= \sum_{ij} p_{ij} \log \frac{p_{ij}}{r_i c_j} \\
&= \sum_{i,j} p_{ij} \log\ p_{ij} - \sum_{ij} p_{ij} \log(r_i c_j) \\
&= -h(P) - (\sum_{ij} p_{ij} \log r_i + \sum_{ij} p_{ij} \log c_j \quad Notice\ \sum_j p_{ij} = r_i,\ \sum_i p_{ij} = c_j \\
&= -h(P) - (\sum_{i} r_i \log r_i + \sum_{j} c_j \log cj) \\
&= -h(P) + h(r) + h(c)
\end{aligned}$$

2.3.1.2 KL Threshold

Now, We want to limit $\mathbf{KL}(P||rc^T)$ to a certain threshold α (by the concavity of entropy) to make a small divergence. By doing so, the entropy of P is still sufficiently large:

$$KL(P||rc^T) \leq \alpha \Leftrightarrow -h(P) + h(r) + h(c) \leq \alpha \tag{2.5}$$

Define again the set of P that satisfies the mapping criteria [2]:

$$U_\alpha(r,c) = \left\{P \in U(r,c) | \mathbf{KL}(P||rc^T) \leq \alpha \text{ or } h(P) \geq h(r) + h(c) - \alpha\right\} \tag{2.6}$$

This makes sure that the set of P in $U(r,c)$ have sufficient entropy with respect to $h(r)$ and $h(c)$.

2.3.1.3 Include Threshold in Minimization

The reason why an upper bound is given to $KL(P||rc^T)$ is to ensure the degree of randomness in P. Retrieve that the value measures how much less entropy P is comparing to the complete randomness or freedom to move. The larger this difference is, the less entropy P will have, the more trips are dependent on one other. The optimal plan we want should not be too restricted but to maintain a minimal degree of freedom for individual trips. Thus we add a threshold to the minimization problem.

Using the notions above, incorporate the entropy constraint α to alter our original problem. The previous $d_M(r,c)$ becomes [2]:

$$d_{M,\alpha}(r,c) = \min_{p \in U_\alpha(r,c)} \langle P, M \rangle \tag{2.7}$$

This result is called *Sinkhorn Distance*. The P has to be selected from the subset with an entropy constraint. The new $d_{M,\alpha}(r,c)$ is a more desired total cost.

For each α, can uniquely find a λ so that

$$d_M^\lambda(r,c) = \left\langle P^\lambda, M \right\rangle, P^\lambda = \operatorname{argmin}_{P \in U(r,c)} \langle P, M \rangle - \frac{1}{\lambda} h(P) \tag{2.8}$$

Each λ gives a solution of of P; as λ approaches infinity, $P^\lambda = \operatorname{argmin}_{P \in U(r,c)} \langle P, M \rangle$

To find the minimum value of $d_M^\lambda(r,c)$, take the derivative with respect to P:

$$\frac{\partial}{\partial p_{ij}}(\frac{1}{\lambda}\sum_{i,j} p_{ij}\log p_{ij} + p_{ij}m_{ij}) = \sum \frac{1}{\lambda}(\log p_{ij} + 1) + m_{ij} = 0$$

$$\begin{aligned} \log p_{ij} + 1 &= -\lambda m\_ij \\ p_{ij} &= \exp(-\lambda m_{ij} - 1) \\ &\propto \exp(-\lambda m_{ij}) \end{aligned}$$

It is clear that the optimal P should be proportional to matrix $K = \exp(-\lambda M)$. The question remains is how to properly scale K to meet the mapping criteria.

2.3.2 Sinkhorn's Theorem

The theorem is stated as the following:

If A is a $n \times n$ matrix with strictly positive elements, then there exist diagonal matrices D_1 and D_2 with strictly positive diagonal elements such that $D_1 A D_2$ is doubly stochastic [10]

Doubly stochastic matrix refers to a matrix whose rows and columns sums to 1.

The theorem also works for non-square matrices.

Applying to our problem, $A = K = \exp(-\lambda M)$, say there are two vectors u and v such that $P^\lambda = diag(u)\exp(-\lambda M)diag(v)$. Then the row sums and column sums of P^λ should match with r and c.

The way to compute vectors u and v is to follow the iterative method in Sinkhorn-Knopp's algorithm:

$u = r/(K * v)$ –compute a row scaling u such that row sums of K matches r

$v = c/(K * u)$ –compute a column scaling v such that column sums of K matches c

Update u and v until convergence [3]

2.3.3 Algorithm

The algorithm follows what is in the paper [2] that has been referring to. It takes the values of r, c, M, λ and gives a result of the total cost and transportation plan associating with λ.

The vector components of mass distribution r and c should be positive. M is scaled by dividing its maximum component, because the entry values are too large for the loop to work. Use x to denote the v which needs to be iterated.

Algorithm 1 Computing d_M^λ with Sinkhorn-Knopp's iteration

Require: r, c, M, choose a λ
Ensure: d_M^λ, P^λ
1: $r = r(r > 0)$; $c = c(c > 0)$; $M = M/max(M)$; $K = \exp(-\lambda * M)$
2: $x = rep(1/r, r)$
3: **while** x does not converge **do**
4: $\quad x = diag(1./r) * K * (c. * (1./(t(K) * (1./x))))$
5: **end while**
6: $u = 1./x$; $v = c. * (1./(t(K) * u))$
7: $d_M^\lambda(r, c) = sum(u. * ((K. * M) * v))$
8: $P^\lambda = diag(u)e^{-\lambda * M}diag(v)$

2.4 Discrete Approach

The discrete approach assumes both the drop-off and pick-up distributions are discrete. We took the same method to work out the two mass distributions: divide a block of NYC equally to smaller blocks, and assign each trip to the closest block centroid. The proportion of vehicles at a drop-off or pick-up hour assigned to a centroid is the mass at that drop-off or pick-up location.

2.4.1 Steps

1, Choose the central area of NYC, longitude [-74.5, -73], latitude [40,41.2]

2, Divide the area into 200x100 blocks equally

3, Take the centroids of those smaller blocks as our possible mapping locations

4, Assign each drop-off & pick-up point obtained from dataset to its closest centroid in terms of Euclidean distance

5, The centroids with non-zero mass become the new drop-off & pick-up locations

6, Calculate the mass for each location by counting the number of trips assigned to the location and diving the total number; obtain two distributions for drop-off & pick-up

7, Calculate the cost matrix for the drop-off and pick-up locations using the function distVincentyEllipsoid, which gives the spherical pairwise distance between drop-off and pick-up locations

8, Choose lambda values and compute the total cost d_M^λ for each

2.4.2 Result

By choosing a grid of λ values, plot the d_M^λ curve:

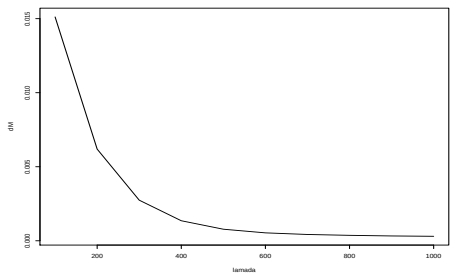

Figure 2.4: Total cost outcome as λ increases

From the previous derivation

$$d_M^{\lambda}(r,c) = \left\langle P^{\lambda}, M \right\rangle, P^{\lambda} = \operatorname{argmin}_{P \in U(r,c)} \langle P, M \rangle - \frac{1}{\lambda} h(P)$$

From the formula we see that the larger λ is, the less $h(P)$ is involved in minimization, hence the result would be closer to only minimizing $\langle P, M \rangle$. In the plot, the cost value is monotone decreasing when λ is starting from some small values, and turned flat at some point. But we should not choose λ to be close to infinity, because in that way $h(P)$ could get large and the plan has high randomness. We want $h(P)$ to be minimized with the total cost in some degree so that the plan is directed or set. λ specifies how much randomness does the plan allow.

On the graph, decide to choose the λ where the curve turns from decreasing trend to flat. Read from the plot gives λ=500.

2.4.3 Plots and Interpretation

To compare the differences of transport plan computed with different λ's, I plotted the plans for λ=500, λ=100, λ=1, λ=0.01.

Each red point represents a drop-off location, and each blue point represents a pick-up location. The gray area is the overlap of many gray edges connecting red and blue points. If there are fewer points, the gray edges are more visible.

I made the optimal P which is the mass transfer matrix a giant vector `masst`, each entry being the mass transfer from one drop-off to another pick-up location. For each edge or individual trip plan, the starting and ending points are the associated drop-off and pick-up locations, while the width of the edge is determined by the value of the corresponding entry in P. The plots below are not distinguishable. However, by looking at the range of `masst` in each plot, we may get some information on the transportation plan.

It is observed that when λ is larger, the associated `masst` values are bigger. It is

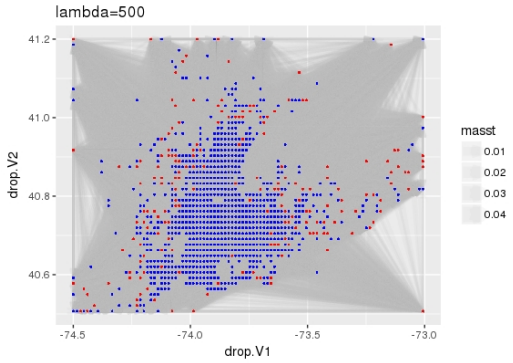

Figure 2.5: Transportation plot $\lambda = 500$

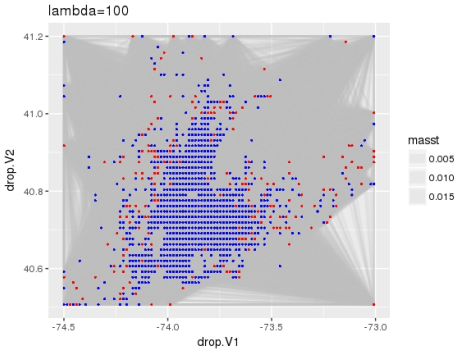

Figure 2.6: Transportation plot $\lambda = 100$

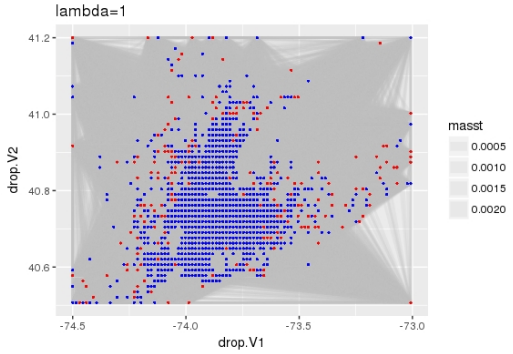

Figure 2.7: Transportation plot $\lambda = 1$

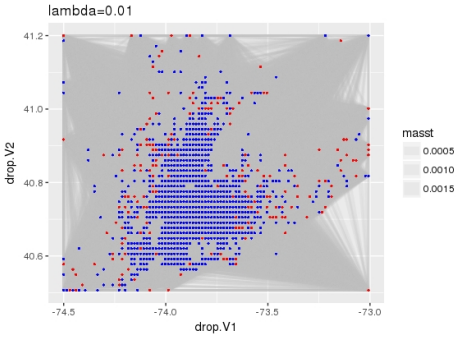

Figure 2.8: Transportation plot $\lambda = 0.01$

because larger λ makes the regularization term $h(P)$ less involved in minimization, we are considering less of entropy, so $h(P)$ is smaller, we have a more directed plan, causing `masst` (the mass transferred from one location to another) have larger values.

2.5 Semi-Discrete Approach

For this approach, assume drop-off distribution is continuous, and pick-up distribution is discrete. So it is a mapping from continuous to discrete space.

The general idea is to sample some drop-off location points from an assumed continuous distribution, and obtain some new pick-up points from the discrete distribution. Then repeat the procedures discussed above to get the optimal transport plan.

2.5.1 Kernel Density Estimation

Since we assume the drop-off location to be a continuous random variable, we want to construct a smooth probability density for it, and sample new data from the density. A natural selection would be Kernel Density Estimator (KDE). It is a non-parametric method

to inference the population distribution based on finite sample points. If $(x_1, x_2, \ldots x_n)$ are samples drawn from a continuous density f, then the kernel density estimator is defined as:

$$\hat{f} = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i), \quad K_h(x) = \frac{1}{h}K(\frac{x}{h}) \tag{2.9}$$

K is the kernel function placed on each individual point. Usually, the standard normal function is used for univariate data. The h is the bandwidth parameter, for the Gaussian kernel case is the variance matrix. The choice of h is flexible, it influences the performance of $\hat{f}$ crucially, so there are many research work done about it. Popular methods used are plug-in selectors, cross-validation selectors, and some adaptive estimation methods such as balloon estimator and pointwise estimator.

The "ks" (stands for kernel smoothing) package in R has the functionality to compute kernel estimators and bandwidth selectors by user's choice, as well as displaying kernel estimators [4]. Since the sample size is large, I chose the plug-in selector, where the bandwidth matrix is assumed constant. It ran much faster than the sample point estimator, where the bandwidth varies with each data point.

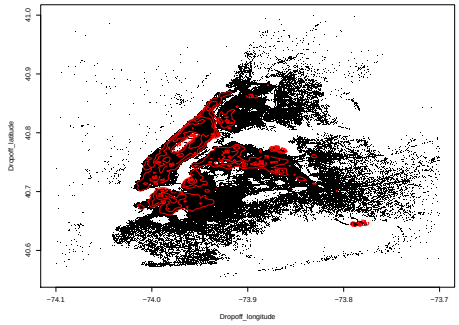

Figure 2.9: Original Dropoff Points and Contour Plots

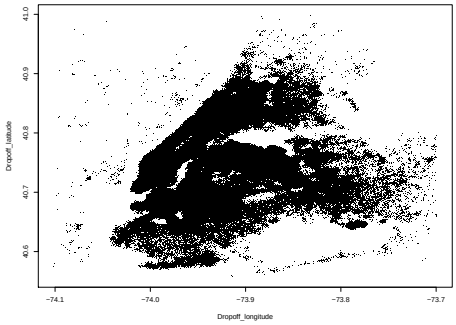

Figure 2.10: KDE Sampled Dropoff Points

The plot on the left is a mapping of all the drop-off location points. The red curves show the 25th, 50th, and 75th percentile of highest density regions, coming from the kernel density estimator. The plot on the right is the mapping of the sampled points drawn from the kernel function. Comparing to the original points, the samples seem to be more clustered around where the contour plot is, and the distribution looks more continuous.

2.5.2 K-Means Clustering

The assumption of pick-up locations are still discrete, but we use a different method to get the mapping locations and associated mass distribution.

In the previous approach, the pick-up location centers are selected manually with equal distances on the range of New York City. This method is simple, but the chosen centers do not represent the most "popular" or "important" pick-up centers. We consider K-means to find the cluster centers that could best represent the distribution of location points.

Since the number of clusters is not known, I ran kmeans for several k values on all the pick-up points, and compute each of their total within-cluster sum of squares.

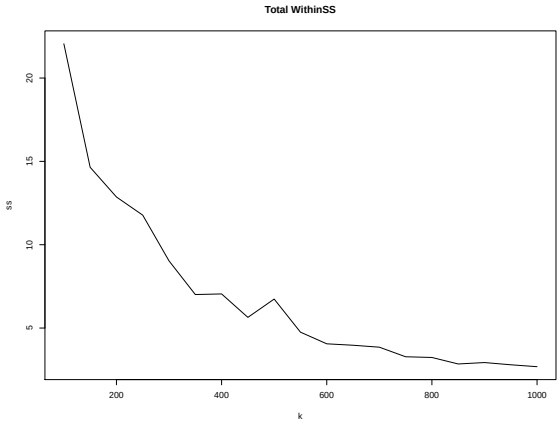

Figure 2.11: Total Within Cluster Sum of Squares in Kmeans

The smaller within-cluster sum of squares means the points are closer to their assigned centroids, so we want to find a not too large k that minimizes it. By the "elbow" method, when k=850 the total `withinss` value is close to the minimum, and the curve is almost flat (see 2.11). So we set K=850 and run kmeans for the pick-up location points. Then take the cluster centers as the new pick-up locations to map to, the proportion of original points assigned to this location as its mass. Now we have a new pick-up density distribution.

2.5.3 Results and Discussion

After getting two new locations and their mass distributions, repeat the algorithm in section 2.3. The results are shown in the following for two different λ.

The two plots below show the result by computing with $\lambda = 1$ and $\lambda = 100$. Again, the red points are drop-off and blue points are pick-up locations, the gray line segments are the mapping directions for each pair of drop-off and pick-up points. The plots only contained 10,000 trips.

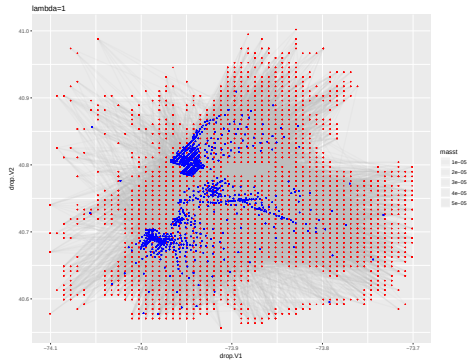

Figure 2.12: Semi-Discrete optimal transport for $\lambda = 1$

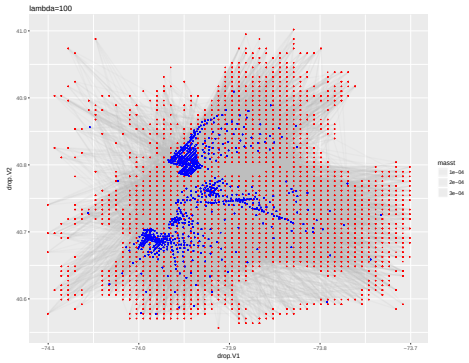

Figure 2.13: Semi-Discrete optimal transport for $\lambda = 100$

The result looks fairly the same as the previous approach, except that there are a significantly smaller number of pick-up (blue) points, since we only chose 850 cluster centers by kmeans. The range of `masst` from the two plots are slightly different. Notice that when λ is larger, `masst` is larger, because the entropy term is less penalized, the algorithm is likely to yield a more "directed" than "random" plan, thus each transportation route carries more mass. The conclusion corresponds to what we got from the previous approach.

CHAPTER 3

Ridesharing

The goal of this chapter is to provide technical principles for real life ridesharing, based on the same dataset described in Chapter 2. The available NYC taxi data has a good amount of trip record information to analyze. I mainly used the green data in August 2015. In the next section, I will talk about the prediction of travel time from one location to another; Section 3 presents a simple searching criteria to find potential rides to pull together, and the results applied on part of the data; Section 4 models distinct shared rides with Poisson, Gaussian, Uniform and Dirichlet distribution, estimates parameters from the original data, and the simulated data.

3.1 Travel Time Prediction

An important concern in ridesharing is that passengers should not wait for too long. To know the waiting time, we need not only know the pick-up and drop-off time for each trip, but need the time traveling from one pick-up location to another. The natural assumption is that the travel time is positively correlated with the trip distance. However, it is also affected by how busy the traffic is, which relates to the time of the day, the location of pick-up, etc.

The goal is to build a regression model to predict the travel time given the available information as accurate as possible. The trip duration is in terms of minutes, calculated by subtracting the pickup time from the dropoff time of each trip. I manually selected 5 relating variables to include in regression: trip distance, longitude/latitude coordinates of pickup and drop-off locations; and I also calculated two predictors: the total number of vehicles (proportion of rides in the hour) and the average speed (miles per minute) in each hour. My intuition is that the location to pick-up and drop-off passengers significantly affect the travel time, since the road conditions in different areas could vary; some areas tend to be busier and have more traffic, so vehicles could not move as fast as on the clear road. The time of the day also plays a role in determining the travel time in two aspects. The more

vehicles traveling at the same time, the longer each trip is expected to last; the average speed in a particular hour directly impacts the travel time. Trip distance is for sure a significant predictor, usually the longer distance to travel, the more time it takes.

3.1.1 Box-Cox Transformation

The first attempt was to build a linear regression model with all the proposed variables. The condition of taking the Ordinal Least Squares approach includes that the response variable follows the Gaussian distribution, and the error variance remains consistent. However, in the real data, trip duration distribution is skewed to the right (see 3.1), which violates the Gaussian assumption. I applied a power transform to convert trip duration to normally distributed.

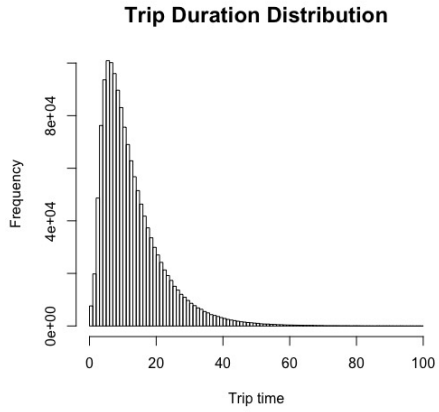

Figure 3.1: Histogram of original trip duration

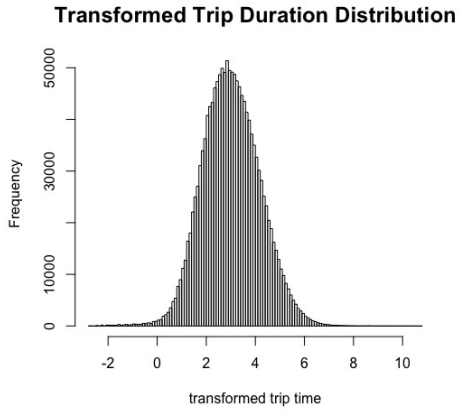

Figure 3.2: Box–Cox transformed trip duration

In linear regression, Box-Cox power transformation is often used to modify the distributional shape of the response variable so that the residuals are more normally distributed. The general form is [1]:

$$y^{(\lambda)} = \begin{cases} \frac{y^{\lambda}-1}{\lambda}, & if \quad \lambda \neq 0 \\ \log y, & if \quad \lambda = 0 \end{cases}$$

y^{λ} is the vector of transformed observations, and the parameter λ is chosen by maximum log-likelihood. The response y is required to be positive, and in this case well satisfied. Here the selected $\lambda = 0.04$. From 3.2, see that after transformation, the response variable trip duration is approximately normal.

Take the transformed trip duration as a response variable, with variables discussed above

to fit a linear model. I also added interaction terms between pick-up longitude and latitude, drop-off longitude and latitude. The result suggests that all the variables are statistically significant. However, I also found that with the same response, no matter what linear model to fit by including any irrelevant predictors, the summary statistic gives significantly small p-values. Usually I rely on the p-value to determine which variables are important, but in this case, I suspect the p-values have the same functionality. It might be due to the fact that the response variable is sufficiently normal. I am not sure here and did not proceed regression with Box-Cox transformed response.

3.1.2 Some Regression Models

I tried some regression methods to model trip duration. To compare their performance, I randomly split the data into 80% training set and 20% testing set. Fit the models using training data, and make a prediction on the test data, then calculate the test Mean Square Error for each model.

3.1.2.1 The Linear Model

The first step was to fit a multiple linear regression model with all the variables described above. When fitting the model, I added two interaction terms between the pick-up location coordinates and between the drop-off coordinates, since each pair of coordinates, in reality, represents one variable. The response variable is still the before-transformation trip duration, in order to compare the test error with other methods.

The result shows that all the predictors included are significant (with p-values < 2e-16), and the adjusted R-squared achieved 0.6781. The model fit is considered well.

3.1.2.2 KNN Regression

Then I considered KNN (K Nearest Neighbour) regression. KNN predicts the response by choosing the closest k data points in the feature space and take the average of their response. The intuition of choosing KNN is that the closer the features are to one other, (i.e. pick-up locations are close geographically and travel distances are similar), the more likely the trips cost the same time.

The question is how to choose the optimal parameter k. When k is too small, the model is too complex and could cause the problem of overfitting. When k is large, each observation is trying to find too many points to take an average, the computational cost would be too large.

The variables to fit KNN model are the same in the linear model. Usually, people pre-process the data by centering and scaling before fitting KNN, in order to eliminate the effect of different measure units of features. Here centering and scaling are not done, except that I manually scaled up the proportion of rides by 10 to make the value be between 0 and 1, which is on the same scale as speed. I believe the raw Euclidean distance would roughly represent the difference between two observations in the data in terms of estimating the trip duration. Even though the location variables are not scaled, it makes sense to apply Euclidean distance directly on them. The trip distance has a wider range than the other two variables (proportion of trips and average speed), which means the difference of this variable in the model would be more dominating, but this variable is also considered more determining in the prediction of trip duration.

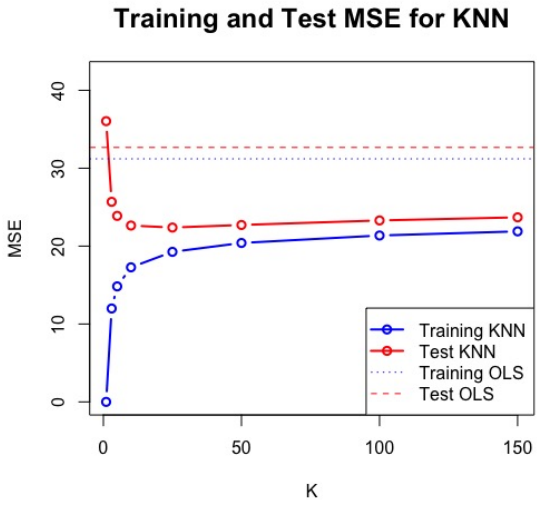

Figure 3.3: Mean Squared Error of training and test data when fitting KNN for different k

In 3.3, The blue curve shows training MSE as k increases, while the red curve shows the change of test MSE. The blue and red broken lines are the MSE of the linear model. Small k gives very small training error and large test error; as k increases, the two errors get closer and closer, eventually converge to an error smaller than the OLS MSE. It suggests that the performance of KNN model beat the linear model.

3.1.2.3 PCR and PLS

In the linear model, the dimension of predictors is 7. It is a little bit large for regression and the coefficients are not so stable. I considered applying Principle Component Analysis (PCA) to the predictors. The technique could well reduce the dimension of original data, while keeping most of the information.

The variables are not scaled for the reason above. PCA results suggest that the first component already captured most of the variance in the data, so choosing the number of components 1 to fit a Principle Component Regression model, and predict on the test set.

Another regression model tried was Partial Least Squares (PLS). This method uses information in the response variable to determine the principal component direction, therefore the selected PCs could be more useful in prediction.

The optimal number of components is selected by finding the minimum cross-validation error. It turns out the cv errors are quite similar to 1 through 7 components. Apply the PLS model to the test data with 1 component, and calculate the MSE.

3.1.3 Comparing Results

The following table shows the test MSE of the four models attempted. The KNN error is when the number of neighbors $k = 25$.

Table 3.1: Test MSE for Different Models

Model	Test MSE
Linear	32.66209
KNN	25.65012
PCR	35.03213
PLS	35.02993

According to the table 3.1, the test MSE for K-Nearest-Neighbour is the smallest. Both PCR and PLS are based on the linear model, but with some variable selection or variable transformation techniques. It makes sense that the error of those three methods is close. The linear model still outperformed the other two, which means dimension reduction is not very helpful for prediction; there might be some correlation among variables, so adding interaction terms could help.

In the following analysis, I chose to use KNN to predict trip duration.

3.2 Dynamic Ridesharing Problem

This section describes a searching mechanism that selects pairs of trips to pull together in order to save time and travel distance. The basic idea for two trips to go along is that both their travel windows and trip routes have large enough overlap. Unfortunately, the data does not tell us the exact trip routes they take. We may only assume the route to be a straight line from the pick-up to the drop-off location.

The proposed searching mechanism borrows the idea of dynamic ridesharing problem defined as follows: "given a fixed number of taxis traveling on a road network and a stream of queries (i.e. a sequence of queries in ascending order of their birth time), we aim to serve each query in the stream by dispatching the the taxi which satisfies with minimum additional incurred travel distance on the road network" [5].

I modified the problem by finding some "candidate" taxis for a particular query first, based on their geographical locations and birth time, then select the one that minimizes the passenger waiting time among all the candidate taxis. After pulling all the pairs of selected trips, calculate how much trip distance is saved with under the ridesharing circumstance. This modification takes the passenger waiting time into more account. The additional incurred travel distance will be constrained by the starting and ending locations of the taxi.

3.2.1 Ride Searching Procedures

For this data set, I treated each trip record as a new query or passenger request, and all other trips as potential taxis to share a ride with. For simplicity, consider the maximum number of rides to be shared is 2. Assume the pickup time is when the query was made, the pickup and dropoff time for each trip is its starting and ending time. Before the searching, sort all the taxi records chronologically by their pick-up time. Take a subset of the data, i.e., a set of trips T that start within the same hour, and apply steps as follows:

1. For each trip (query) indexed by i in T, select the set of trips that show up (start) after but no later than 5 minutes of its pick-up time t_0;

2. Among all the trips selected by step 1, find the subset of trips that start and end within 500 meters around the starting and ending locations l_s, l_e of trip i respectively;

3. From all the trips selected by step 2, select trip j that minimizes t_w, the waiting time of the query i.

$$t_w = t_r + t_p - t_0 \tag{3.1}$$

Where t_w is how long the trip (query) i needs to wait to join another trip, t_r is the start time of each targeted trip, t_p is how long for this trip to travel to l_s, predicted by the linear model described in section 2.2.2, and t_0 is the pick-up time of trip i or the birth time of the query.

4. Pull trip i, trip j together, and take both of them out of the set T. Iterate above from trip $i + 1$.

In figure 3.4, the two centers of the circles represent the pick-up and drop-off locations

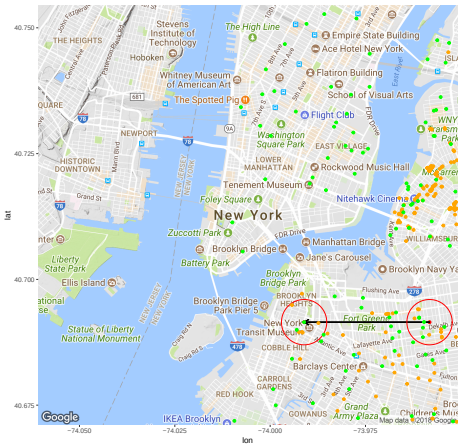

Figure 3.4: A trip instance on NYC map with radius 500m circle around starting and ending locations

for a random trip. The arrow is pointing to the drop-off point. Each red circle is centered at the pickup/dropoff point, with radius 500 meters. The green points are the pickup points of proceeding trips, while the orange points are the dropoff points of proceeding trips. The figure illustrates the idea of selecting trips that start and end within a certain distance to the query. We would want to find trips that the corresponding green point fall in the circle on the right and orange point in the circle on the left. The two has to be satisfied at the same time.

3.2.2 Reduced Trip Distance

Given a sequence of taxi rides, we can apply the searching criteria described above, and find pairs of rides that can be shared or combined together. Based on the result, calculate how much travel distance is reduced after sharing.

For each pair of proposed sharing trips r_1, r_2, calculate the following distance:

$l_{1,1}$: the distance between the pick-up and drop-off location of r_1;

$l_{2,2}$: the distance between the pick-up and drop-off location of r_2;

$l_{1,2}$: the distance between the pick-up location of r_1 and the drop-off location of r_2;

$l_{1,2,p}$: the distance between the pick-up location of r_1 and $r2$;

$l_{1,2,d}$: the distance between the drop-off location of r_1 and $r2$;

If the rides are not shared, r_1 and r_2 are going separately, and the total distance of them would be

$$l_{noshare} = l_{1,1} + l_{2,2} \tag{3.2}$$

But if combining the two trips together, we want to find the shortest path possible. Since

we treat r_1 as a passenger request, it does not move until the other ride comes, let r_2 first drive to meet r_1 at its starting location. Then choose the smaller one of $l_{1,1}$ and $l_{1,2}$ to decide driving to which of the destinations first. After dropping off one of them, drive to another destination. The total distance would be represented as:

$$l_{share} = l_{1,2,p} + min(l_{1,1}, l_{1,2}) + l_{1,2,d} \tag{3.3}$$

Then for each pair of shared rides, the travel distance saved is

$$l_{save} = l_{noshare} - l_{share} \tag{3.4}$$

3.2.3 An Example

I did an experiment with all the trips starting between 8:00 am and 9:00 am on the same day. There are 1175 trips occurred in total. Eventually, 46 pairs of trips are selected to pull together. The total distance of these trips without ridesharing is 263738.6 meters, and the new total distance traveled is 153376 meters. 41.8% of the travel distance is saved.

The result proves that ridesharing does have an effect of reducing travel amount, therefore saving more energy and generating less pollution.

3.3 Modeling Shared Rides and Individual Trips

This section introduces a way of statistical modeling for ridesharing problem. The general idea is to make model assumptions on the existing data, then estimate parameters from a frequentest approach. Plugging the parameters into the model, and simulate new data, then estimate the parameters again from the simulated data, compare those new parameters to the previous ones, see if the model fits data well.

3.3.1 Model Introduction

Assume all the shared rides is a cluster on the space of NYC Θ, with features pickup/dropoff time, pickup/dropoff locations. We can model the number of cluster centers k by Poisson distribution, with parameters λ_1 and Θ:

$$k \sim Poi(\lambda_1 vol(\Theta)) \tag{3.5}$$

λ_1 represents the number of trips per unit area. Then conditioning on the number of shared rides k, we can set a uniform prior distribution for the k pick-up locations $\theta_{1p}, \theta_{2p}, \ldots \theta_{kp}$ and pick-up time $t_{1p}, t_{2p}, \ldots t_{kp}$. Suppose the range of pick-up time is from T_1 to T_2:

$$\theta_{1p}, \theta_{2p}, \ldots \theta_{kp} | k \sim \mathcal{U}(\Theta) \tag{3.6}$$

$$t_{1p}, t_{2p}, \ldots t_{kp} | k \sim \mathcal{U}(T_1, T_2) \tag{3.7}$$

Notice that in 3.7, it is discrete uniform distribution, since time is a discrete variable measured by seconds.

Given a pick-up location $\theta_{jp} \in \{\theta_{1p}, \theta_{2p}, \ldots \theta_{kp}\}$ and pick-up time $t_{jp} \in \{t_{1p}, t_{2p}, \ldots t_{kp}\}$ from one of the clusters, we can model the conditional drop-off location θ_{jd} and drop-off time t_{jd}. Since the drop-off time is restricted to be later than pick-up time, the difference between drop-off and pick-up time is only going to be positive, so we model it by Exponential distribution with parameters λ_2. The drop-off location will be modeled as a Gaussian, which is centered at the pick-up point θ_{jp} with variance Σ_1:

$$\theta_{jd} | \theta_{jp} \sim \mathcal{N}(\theta_{jp}, \Sigma_1) \tag{3.8}$$

$$t_{jd} | t_{jp} \sim \text{Exp}(\lambda_2) + t_{jp} \tag{3.9}$$

Now we have k cluster centers, and assume each cluster has a probability $\pi_1, \pi_2, \ldots \pi_k$, which follow Dirichlet distribution, with parameter $\alpha_1, \alpha_2, \ldots \alpha_k$

$$\pi_1, \pi_2, \ldots \pi_k | k \sim Dir_k(\alpha_1, \alpha_2, \ldots \alpha_k) \tag{3.10}$$

Each α_i corresponds to the probability of the i th cluster center π_i. Given a cluster center, we can sample the pick-up, drop-off locations and times for the individual trips assigned to it. Suppose there are n taxi trips. For each ride $i \in \{1, 2, \ldots n\}$, first sample a cluster center z_i it could belong to:

$$z_i \sim Dir_k(\alpha_1, \alpha_2, \ldots \alpha_k) \tag{3.11}$$

Then from the distributions described above, with the specified cluster z_i, we can obtain

the pick-up location θ_{z_ip}, drop-off location θ_{z_id}, pick-up time t_{z_ip} and drop-off time $t_{z_i}d$. These variables serve as the mean of Gaussian distribution to sample the trip pick-up location x_i, drop-off location y_i, pick-up time t_{pi}, and drop-off time t_{di}. Note that drop-off time has to be greater than pick-up time, so use a truncated normal for that simulation.

$$x_i \sim \mathcal{N}(\theta_{z_ip}, \Sigma_{z_1}) \tag{3.12}$$

$$y_i \sim \mathcal{N}(\theta_{z_id}, \Sigma_{z_2}) \tag{3.13}$$

$$t_{pi} \sim \mathcal{N}(t_{z_ip}, \sigma_1^2) \tag{3.14}$$

$$t_{di} \sim \mathcal{N}(t_{z_id}, \sigma_2^2)\mathbf{1}\{t_{di} > t_{pi}\} \tag{3.15}$$

3.3.2 Parameter Estimation

The first step is to get some trip observations during a time interval, apply kmeans to find their cluster centers. I used all the trips that started during 8-9am on 08/01/2015. Since the number of clusters k is required to specify in kmeans, I tried a variation of k values between 100 and 500, and selected an optimal one via gap statistic. A method described by [11] is to compute statistic:

$$CH(k) = \frac{B(k)/(k-1)}{W(k)/(n-k)} \tag{3.16}$$

Where $B(k)$ and $W(k)$ are between- and within-cluster sum of squares, for k clusters. The maximum of $CH(k)$ associated with the best k.

The data suggests that the best k value is 250. The volume of Θ is approximated by the area of New York City that green taxis could appear at. By 3.5, the maximum likelihood estimation of $\lambda_1 vol(\Theta)$ is the mean of cluster numbers k, in this case 250. Then get the estimated result:

$$\lambda_1 = \frac{k}{vol\Theta} = 1.021117 \tag{3.17}$$

Then following 3.6 and 3.7, I generated 250 pickup locations and times from the uniform distribution. For simplicity, the pick-up locations are sampled from the 8-9am trips in September. T_1 and T_2 are the earliest and latest pick-up time in the original trip data used for clustering.

Then the drop-off locations are obtained by sampling from the Gaussian distribution,

centered at each corresponding pick-up location. The covariance matrix Σ_1 is estimated by the pick-up locations. The following figure shows the sampled locations points. Each plot contains 250 points which are the number of clusters.

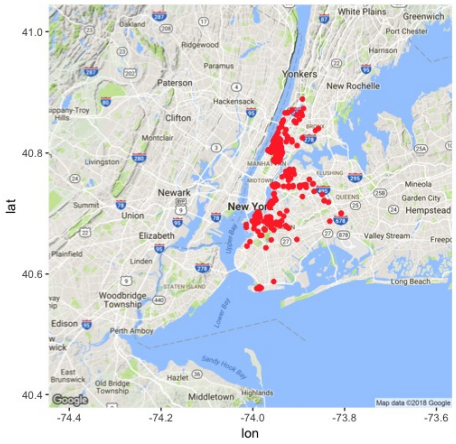

Figure 3.5: Sampled Uniform Pickup Points

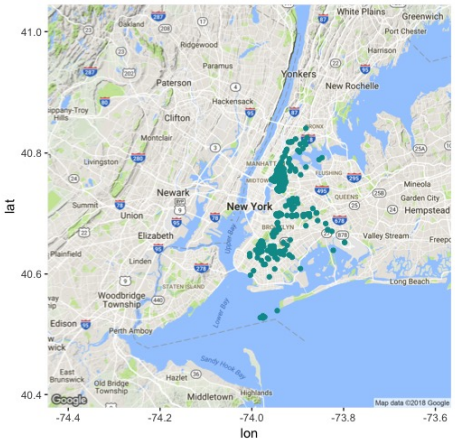

Figure 3.6: Sampled Gaussian Dropoff Points

To sample the drop-off time, I first estimated the exponential parameter λ_2. It is modeling the difference between the drop-off and pick-up time which is trip duration, so the parameter is the reciprocal of the trip duration (in minutes) mean:

$$\lambda_2 = \frac{1}{mean(tripdur)} = 0.07813695 \tag{3.18}$$

The next step is to estimate the Dirichlet parameters. In Minka's paper [6], it describes a method to estimate all the α_k in terms of moments of the data:

$$E[\pi_k] = \frac{\alpha_k}{\sum_k \alpha_k} \tag{3.19}$$

$$\sum_k \alpha_k = \frac{E[\pi_1] - E[\pi_1^2]}{E[\pi_1^2] - E[\pi_1]^2} \tag{3.20}$$

π_k is the probability of a single trip being assigned to cluster k, and can be estimated by the proportion of trips in the kth cluster. In order to obtain the first and second moments of π_k, I generated over 100 sets of $\pi_1, \pi_2, \ldots \pi_k$. I extracted 8-9am trips on each day of August 2015, each time randomly sample 500 trips on the same day, assign them to the closest one of the 250 cluster centers, then calculate the proportion of assigned trips in each cluster as

π_k. The time variables in each sampled data are processed to be around the same range of time as the cluster centers. By formula 3.20 the result:

$$\sum_k \alpha_k = 570.9403 \tag{3.21}$$

And the 250 individual α_k are calculated by 3.19.

3.3.3 Simulate Data and Compare Parameter Estimation

From the probability distribution of cluster centers, individual trip data (pick-up time/location, drop-off time/location) could be simulated from normal and truncated normal density, as described in Section 3.3.1. Then repeat the procedures in Section 3.3.2: apply kmeans to choose the optimal cluster number $k' = 500$; calculate the mean of trip duration in the simulated data; sample 8-9am trip records through the month, assign them to the 500 new cluster centers and calculate the proportion to estimate the α'_ks. Results of the newly estimated parameters:

$$\lambda'_1 = \frac{k'}{vol\Theta} = 1.731122 \tag{3.22}$$

$$\lambda'_2 = \frac{1}{mean(tripdur')} = 0.07901255 \tag{3.23}$$

$$\sum_k \alpha'_k = 669.2394 \tag{3.24}$$

The parameters estimated from simulated data are similar to but not very close to the parameters got previously. This suggests that the model for the trip data makes sense to a certain degree, but is not a great fit. Ways of improvement are discussed in the next chapter.

CHAPTER 4

Conclusions and Future Outlook

4.1 Conclusions

In the Optimal Transport chapter, the two approach (discrete and semi-discrete) eventually give pretty similar results. The difference in visualization is due to the choice of locations points to plot. The agreement of two approaches might be an indication that the Sinkhorn's algorithm could provide valid solutions to the optimal transport problem. It is also possible that it does not matter which distribution the data points of interest come from, because the key steps to solve the problems are exactly the same.

In the Ridesharing chapter, one trivial conclusion is that after searching for the rides that start and end at the same time and around the same area, combining them would actually save some travel cost. The statistical modeling of shared rides is reasonable, and could be applied for practical means if the model is additionally perfected.

4.2 Limitations

4.2.1 Sampling Uniform Pickup Points

The pick-up locations of clusters should be uniformly distributed on the New York City space. Instead of doing uniform sampling, I randomly sampled points from another month data. However, the actual distribution of pick-up locations is not uniform, some areas have a distribution of higher density than others. This approach is not valid.

To draw uniform samples, the difficulty is to specify the boundary of the green taxi pick-up area, since the area is not a regular geometry.

4.2.2 Sampling Drop-off Points Conditional on Pickup

In 3.8, I used the covariance in the pick-up points as the covariance matrix for this Gaussian density to sample pick-up points. It is very unlikely to be accurate. Initially, the following relation is suggested to me:

$$E\left\|x-y\right\|^2 \sim \sigma^2\chi_2^2 \tag{4.1}$$

Where x and y are pick-up and drop-off locations, σ^2 is the variance associated with all the trips whose pick-up and drop-off are in (x, y). The expectation of the $l-2$ norm of trip distance follows the chi-square distribution with a multiplier σ^2. If x and y are the location pairs within each cluster, then the variance of drop-off location given pick-up location is estimated as:

$$\sigma^2 = \frac{\left\|x-y\right\|^2}{2n} \tag{4.2}$$

Then use σ^2 multiple by 2x2 identity matrix as the covariance matrix Σ in 3.8, so each cluster has a Σ_j. However, when I apply this variance estimation, the resulted drop-off points are mostly in the water area on NYC map, which is not desired. I need a better way to make sure the simulated points are within space Θ. For the current analysis, I discard this approach. Using the variance from pick-up points at least gives the sampled points on land.

Another problem came to me is that even though the sampled drop-off points are on land, they do not represent the true drop-off area.

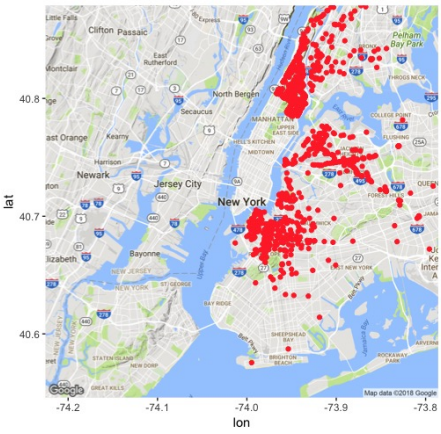

Figure 4.1: GReal pick-up Points

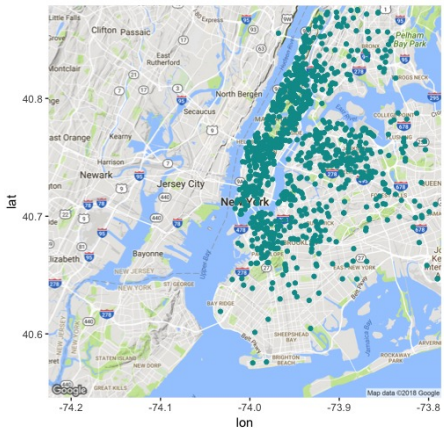

Figure 4.2: Real drop-off Points

The figures shown here are Distribution of Real Pick-up and Drop-off Points. We can

see that the there are no pick-up points in lower Manhattan area, but there appear many drop-off points. If we only sample drop-off points centered at pick-up points, we might not get anything in this area. Then the simulated data does not correspond to the real case. The sampling method to simulate drop-off points needs a big improvement.

4.2.3 Stability of Dirichlet Parameters

In 3.20, it only uses the moments of the first cluster probability π_1. In the paper [6] it says using any other π_k could also estimate $\sum \alpha^k$. But I checked the generated data, and found that the value of $\sum \alpha^k$ varies a lot with different π_k. Now it is a question whether the result I obtained is accurate or not, since it is so unstable.

4.3 Future Directions

4.3.1 Location Coordinates

In the future working with transportation data, I hope to have a better way to treat the 2-dimensional coordinate variable (longitude and latitude) as a one-dimensional variable, since it would be much easier and more convenient for calculations and model fitting. In prediction problem, a location being treated as one variable is more desirable because it contains the full location information, and I expect to have better results in this way.

I found Geohashing to be a good representation of longitude/latitude pairs as geographic "boxes". The direct application of Geohash (i.e. calculate the distance between two points) is not clear to me yet, but I might want to investigate and use more of it in the future.

4.3.2 More Variables and Methods in Trip Duration Prediction

In section 3.1 I only included trip distance, pick-up and drop-off locations, number of trips in each hour and the average speed per hour in the model. But there are some other variables that might be associated with duration of a trip. For example, more passenger count could increase the travel time fixing all other variables, because they might need to be dropped at different locations and the taxi has to spend some time redirecting and traveling extra distances. Payment type could also be influential, since paying in cash takes slightly more time than paying in a credit card. Tip amount is also a potential predictor. If a taxi reaches to the passenger's destination faster than expected, it is possible that the passenger gives more tips. In the future analysis, I would like to include more variables, not limited to the above 3, to be in the prediction model.

I would also like to try some tree-based methods in addition to the traditional regression. I might want to perform bagging, random forest, and boosting. Those methods could also help do some variable selection as I have more variables.

4.3.3 Bayesian Approach for Parameter Estimation

When estimating the parameters in the model for shared rides, I took a frequentest approach and estimated them by maximum likelihood. I am also very interested in taking a Bayesian approach, by applying some MCMC techniques (Gibbs sampling, Metropolis-Hasting). Then I could estimate parameters from one month of data, and use another month of data to test the model fit.

4.3.4 Combine of the Two Problems

Essentially, the purpose of both optimal transport and ride share is to efficiently utilize the available taxis resources to satisfy the demand of passengers. The former deals with the entire taxi trip at a snapshot of time, while the later looks at individual trips by clusters. I am looking forward to working on a combination of the two problems, for example, computing an optimal transport plan for all the shared trips, or pairing up rides to share in the current macro level optimized transport plan.

Bibliography

[1] G. E. Box and D. R. Cox. “An analysis of transformations”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* (1964), pp. 211–252.

[2] M. Cuturi. “Sinkhorn distances: Lightspeed computation of optimal transport”. In: *Advances in neural information processing systems*. 2013, pp. 2292–2300.

[3] Dirk. *Calculating transport plans with Sinkhorn-Knopp*. Dec. 2017. URL: https://regularize.wordpress.com/2015/09/17/calculating-transport-plans-with-sinkhorn-knopp/.

[4] T. Duong and M. T. Duong. “Package ‘ks’”. In: (2018).

[5] S. Ma, Y. Zheng, and O. Wolfson. “T-share: A large-scale dynamic taxi ridesharing service”. In: *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*. IEEE. 2013, pp. 410–421.

[6] T. Minka. *Estimating a Dirichlet distribution*. 2000.

[7] *NYC Taxi & Limousine Commission*. Dec. 2017. URL: http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml.

[8] N. Y. S. E. Research and D. Authority. *New York State Greenhouse Gas Inventory: 1990 – 2014*. Research rep. New York State Energy Research and Development Authority, 2017.

[9] L. Rüschendorf. “Monge-Kantorovich transportation problem and optimal couplings”. In: *Jahresbericht der DMV* 3 (2007), pp. 113–137.

[10] R. Sinkhorn. “A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices”. In: *The Annals of Mathematical Statistics* 35.2 (June 1964), pp. 876–879.

[11] R. Tibshirani, G. Walther, and T. Hastie. “Estimating the number of clusters in a data set via the gap statistic”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 63.2 (2001), pp. 411–423.

[12] *U.S. Energy Information Administration*. July 2017. URL: https://www.eia.gov/state/seds/data.php?incfile=/state/seds/sep_sum/html/rank_use_source.html&sid=US.