Optimal Transport Applied to Commuter Subway Ridership

Sam Royston

Abstract

We define a Monge-Kantorovich type problem on the NYC subway system to better understand commuter behavior and it's implications for congestion. NYC subway turnstile entrance and exit data is used to infer a cost minimizing commuter flow between each station, analogous to a trading network where entrances are 'production' and exits are 'consumption'. We introduce and evaluate modifications to the model specifically for this problem and use optimal transport to estimate which segments of the system are most heavily used and to evaluate a selection of proposed transit plans.

Introduction

New York city is home to the world's largest mass transit rail system; the NYC subway has 422 stations and hundreds of miles of track. The Metro Transit Authority (MTA) has made public a variety of real time and static data feeds regarding the usage of the system, however to the author's knowledge none have been used to estimate transit flows using an optimal transport model. Optimal transport models can provide insights into rider's choices, congestion patterns, and enable us to make specific queries on things like line usage and transfer frequency. In this paper, we describe the typical optimal transport model applied to trading networks and how to modify it so that it is applicable to an urban rail system like the NYC subway. Furthermore, we explore the results of the model applied to real data provided by the MTA, and discuss implications for commuters and transit planners.

1 Optimal Transport Networks

1.1 Trading Networks

The canonical setting for optimal transport is a trading network for a single resource with a set of supply and demand nodes $\mathcal{X}$ (cities), and a set of directed arcs $\mathcal{A}$ (roads) between them. The primal and dual problems in this settings are ones of transport cost minimization (optimizing trading flow rates over all conduits) and profit maximization (optimizing prices for goods at each city). An $|\mathcal{A}| \times |\mathcal{X}|$ edge-node matrix ∇ defines the topological structure of this system. For each $x \in \mathcal{X}$ and $a \in \mathcal{A}$, we have and entry in ∇:

$$\nabla_{ax} = \begin{cases} +1, & \text{if } a \text{ is an in-edge of } x \\ -1, & \text{if } a \text{ is an out-edge of } x \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Note that ∇ is sparse by definition, containing only two entries per row (1 and -1 for each edge). The production (if positive) or demand (if negative) of each city is described by the vector $\mathbf{n}$, with one entry for each city. Additionally we describe the costs associated with each arc as a vector of edge weights $\mathbf{c}$. As mentioned, the primal and dual problems can be used to infer *flows* over edges $\mathbf{\Pi}$, by solving

$$\min_{\mathbf{\Pi}_a \geq 0,\ \forall a \in \mathcal{A}} \mathbf{\Pi} \cdot \mathbf{c}\,, \quad \text{subject to} \quad \nabla^\top \cdot \mathbf{\Pi} = \mathbf{n} \tag{2}$$

or *prices* $\mathbf{\Phi}$ at each city by solving

$$\max_{\mathbf{\Phi} \in \mathbb{R}^{\mathcal{X}}} \mathbf{\Phi} \cdot \mathbf{n}\,, \quad \text{subject to} \quad \nabla \cdot \mathbf{\Phi} \leq \mathbf{c} \tag{3}$$

In order for the above optimization problems to be feasible, a few criteria must be met, each with an intuitive interpretation.

1. **Balancedness:** Nodes are defined as supply or demand nodes based on the sign of their entries in $\mathbf{n}$, and we assume that exactly all of the supply is eventually consumed by the demand nodes, i.e.

$$\sum_{x \in \mathcal{X}} n_x = 0$$

2. **Connectedness:** The set of supply nodes $\mathbf{n}_+, \subset \mathbf{n}$ and demand nodes $\mathbf{n}_- \subset \mathbf{n}$ are such that $\mathbf{n}_+$ is strongly connected to $\mathbf{n}_-$, i.e. there is a path from every $n_i \in \mathbf{n}_+$ to every $n_j \in \mathbf{n}_-$.

3. **No profitable loop**: There are not arbitrage opportunities with negative cost loops. We can enforce this easily by setting

$$c_a \geq 0, \ \forall c_a \in \mathbf{c}$$

These are the basic components of the optimal transport problem defined on trading networks. Various modifications must be made to better accommodate the commuter transit setting.

1.2 Optimal Commuter Flow in Urban Rail Networks

There are a many inconsistencies between the idealized trading network model and the reality of urban rail transit. While we enumerate these inconsistencies in full later, in this section we describe modifications made to the trading network model mentioned above to better fit a mass transit rail system. In this section, we focus on the flow determination problem (2).

Net Production and Demand

In the trading network model described in section 1.1 each vertex $x \in \mathcal{X}$ is associated with a single net resource contribution n_x, meaning each vertex is either exclusively a supply or a demand node, but never both. Intuitively, in a trading network this means that all the bread produced at city x will also be consumed by city x if the demand for bread is greater than or equal the production. However in our setting this is not the case, as each station is endowed with both commuter production rates $\mathbf{p}$ (entrance counts) and commuter *relief* rates $\mathbf{r}$ (exit counts). The analog in the prototypical trading model described in section 1.1 is to take the differences between these two values for each city and use them to populate the demand vector $\mathbf{p} - \mathbf{r} = \mathbf{n}$. As mentioned earlier, this yields each station as a production or demand node based on the sign of this difference. However, using the net-value for entrances and exits seems inappropriate since no one enters a subway station with the intention of exiting that same station without setting foot on a train. The trading model fails to make use of the extra data provided by the entrance and exit numbers and must treat busy stations with entrances and exits that net to zero the same as as station with absolutely no usage. To address this we introduce additional constraints which make use of both the entrance and exit rates. We define new matrices ∇_+ and ∇_- that consist of the positive and negative components of the edge-node matrix ∇, i.e. separate matrices marking in-edges and out-edges.

$$\nabla_{+_{ax}} = \begin{cases} 1, & \text{if } a \text{ is an in-edge of } x \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

$$\nabla_{-_{ax}} = \begin{cases} 1, & \text{if } a \text{ is an out-edge of } x \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

In the modified model the flow vector $\mathbf{\Pi}$ must also satisfy

$$\nabla_-^\top \cdot \mathbf{\Pi} \geq \mathbf{p}\ , \ \ \nabla_+^\top \cdot \mathbf{\Pi} \geq \mathbf{r} \tag{6}$$

and as before, the conservation of mass equation still holds

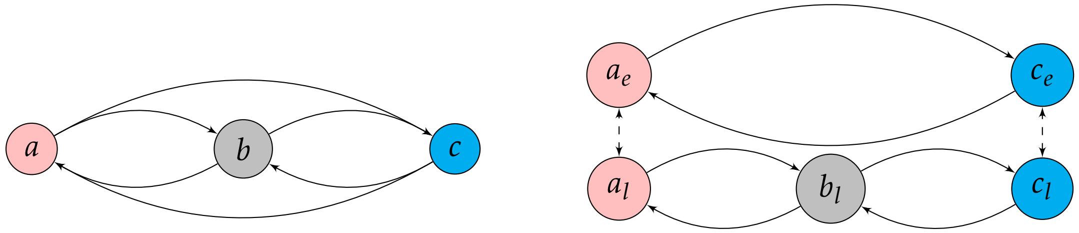
$$\nabla^\top \cdot \mathbf{\Pi} = \mathbf{n} = \mathbf{p} - \mathbf{r} \tag{7}$$

The interpretation here mirrors real life precisely: each commuter who enters a station can be expected to travel along exactly one of the edges emanating from that station, thus the sum of the total outward flow from that station must be greater or equal to that station's entrances (in general it will be more due to surplus flow into the station).

Commute Distance

In this system there are other factors at play besides efficiency. In fact, commuter transport my not be optimal at all. The commuter is not simply trying to minimize the cost of their commute and there are likely other factors at play; such as the pay offered by jobs in a certain neighborhood and the desirability of certain residential areas. We can say with confidence that commuters do not commute a single stop, therefore we might be inclined to place a lower bound on the flow at each arc. This does not change the behavior of the system much in practice and will often simply result in the arcs with zero flow being assigned the lower bound. In order to promote bidirectional travel between each station while allowing flexibility in allocation we can put a lower bound on the bidirectional flow between two stations. To do this we can use Ξ, an $|\mathcal{A}| \times |\mathcal{U}|$ matrix consisting of zeros and ones, setting $\mathcal{U}$ to be the set of non-directed arcs on our network such that $|\mathcal{U}| = \frac{|\mathcal{A}|}{2}$, where $u_{xy} \in \mathcal{U}$ if and only if $a_{xy}, a_{yx} \in \mathcal{A}$, i.e. a is a permutation of u

$$\Xi_{au} = \begin{cases} 1, & \text{if } a \text{ is a permutation of } u \\ 0, & \text{otherwise} \end{cases} \tag{8}$$



Transit network of 3 stations with one express line and one local line and with no transfer penalty

Augmented network with transfer edges

Figure 1: Network augmentation procedure to model transfer times

for $a \in \mathcal{A}$ and $u \in \mathcal{U}$. And require that for some constant C

$$\Xi^\top \cdot \mathbf{\Pi} \geq C \tag{9}$$

Subway Lines and Transfers

We must modify the network to reflect that each subway line is not identical (in terms of service density) and perhaps more importantly, the cost of transferring lines. To achieve this we create nodes for each line-station pair which form a clique (S, T) fully connected by *transfer* edges T where $|S|$ is the number of lines serviced at the station. The example in figure 2 has two lines e and l (express and local).

Transport Costs

We decompose the transport cost c_{xy} associated with each edge into two components.

$$\mathsf{c}_{\mathsf{xy}} = \omega_x + d_{xy}$$

Wait time: ω_x is the wait time or train arrival frequency for a given station x. $\omega_x = 0$ for all edges except for transfer edges. The wait time incurred by riders who have just entered the system is neglected for simplicity.

Distance: d_{xy} is the travel time between stations x and y

Ideally, the transfer edges would discourage the system from modeling unrealistic behavior like instantaneous transfers and switching directions seamlessly.

2 Application to Subway Data

2.1 Data Input

The Metro Transit Authority has made available a number of different data feeds, some updated weekly. The software written for this study combines data from four different feed types to derive the network structure and usage data needed for the transport model.

Turnstile Data http://web.mta.info/developers/turnstile.html This data is updated weekly with new turnstile exit and entry counts. Each row in the file is a separate turnstile device which must be referenced against a station by its name. Furthermore, each device registers *cumulative* counts so counts over time intervals must be subtracted out. We used this dataset to determine the per-station values $\mathbf{n}_x$,$\mathbf{r}_x$, and $\mathbf{p}_x$.

Station Data http://web.mta.info/developers/sbwy_entrance.html This data set consists of station locations as well as entrance locations, station names and the lines that service it. This was used to corroborate subways lines assigned to each station from the turnstile data.

Stops Data The stops data is located in the GTFS schedule data folder of http://web.mta.info/developers/. This file is important because it contains location data and it can be used to link station names to station ids, which might look something like "A19" and are the only consistent form of station identification across the MTA subway data (though they are not included in the Turnstile and Station data mentioned above)

Stop-Times Data This data is what one must analyze to derive the links between each station. It lists the scheduled arrival and departure times for each train for each station. Connectivity between the IDs mentioned above can be determined by looking at successive entries in this file. We also use the data in this file to determine the d_{xy} average trip time between stations.

Our initial goal was to generate aggregate *supply* and *demand* figures describing whether subway stations *produce* commuters or *consume them*. The entrance and exit counts aggregated by each station name are natural choices for this, however we found the exit data to be corrupted in many stations, likely due to the emergency exit door which are often used in crowded stations during rush hour. Since it was not clear how to check

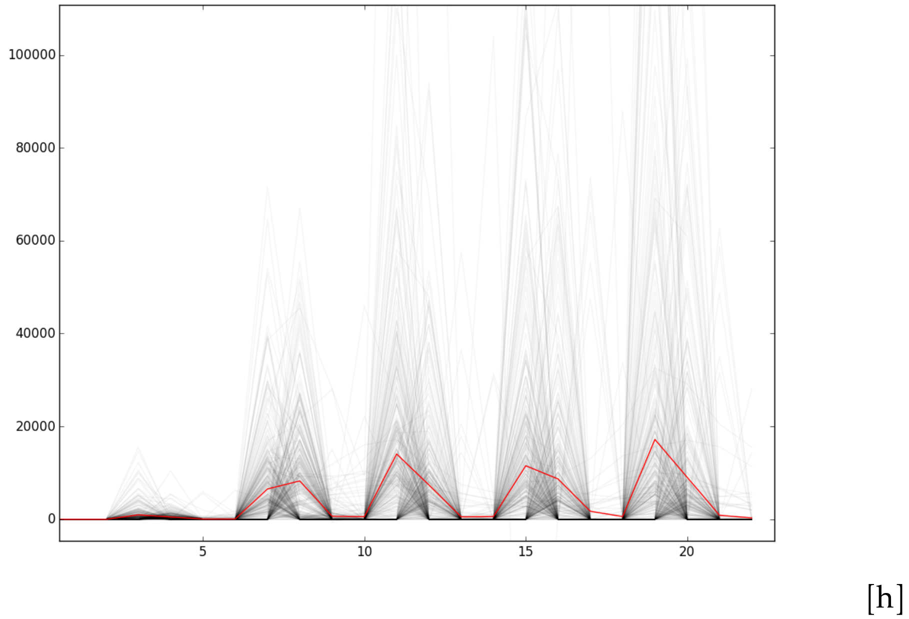


Figure 2: Entrance rate changes over 1 hour intervals throughout an average day for each station (gray lines), and their average (red line). It appears as though there are four times throughout the day when data is collected from the turnstiles.

which stations had a frequently used emergency exist, we instead made use of the fact that entrance counts are time-stamped and measured roughly four times per day, depending on the turnstile. Days were partitioned into two intervals $(0, t), (t, 24)$, written in hours, and entries coming from the first interval were interpreted as true station entries, while the evening entries were counted as morning exits. This technique assumes that the dominant majority of subway users are commuters, and to ensure that this was likely the case we restricted the data-set to weekdays only.

We manually selected t so that the number of entries in either interval would be roughly the when summed over all stations. The turnstile data, station data, and stops data are inconsistent in the number of stations, station names, and station lines and therefore present a record linkage challenge. We used ad-hoc approximate matching techniques to coalesce the data into files of vertices and edges. In our experience with this dataset, the most effective string metric for measuring similarity between station names was the Jaro-Winkler distance[9][**?**]. The Jaro-Winkler scores based on prefix similarity as well as the edit distance based components of the Jaro distance.

$$m = chars(s_1) \cap chars(s_2)\ , \quad t = \text{number of transpositions}$$

$$d_{jaro}(s_1, s_2) = \begin{cases} 0, & \text{if } a \text{ is an in-edge of } x \\ \frac{1}{3} \cdot \left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m-t}{m} \right), & \text{otherwise} \end{cases}$$

$$d_{jarowinkler}(s_1, s_2) = d_{jaro}(s_1, s_2)\ + p \cdot \ell \cdot (1 - d_{jaro}(s_1, s_2))$$

Measuring the distance in meters between known stations to identify those that should have transfer edges was quite effective when combined with filtering for line overlap via the Jaccard similarity.

2.2 Meeting the Assumptions

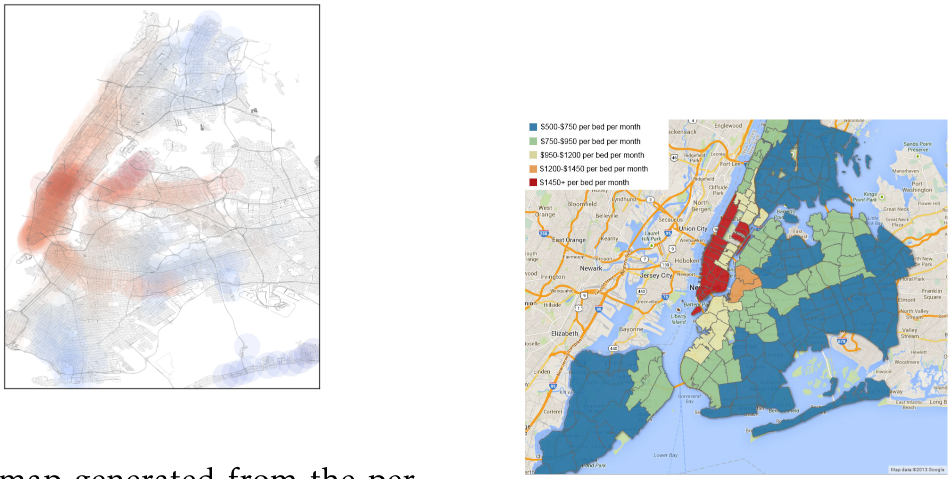
In section 1.1 we defined a few assumptions that must be met for the problem to be feasible. Balancedness is not immediately satisfied by the real-world data and determines our choice of the partitioning time t. After selected t, we must still ensure that morning entrances and evening entrances are perfectly balanced by randomly inserting artificial entries into the deficient partition. In addition, the connectedness criterion is not immediately met by the data, and one must ensure that the appropriate stations are connected via transfer edges.

2.3 Station Price

The dual problem (3) has a less obvious interpretation when applied to a transit rail system which we explore in figure 3. We infer optimal prices $\mathbf{\Phi}_x$ associated with each station, what but precisely what these prices should refer to in this case is unclear, however they have a noticeable similarity to reported rents.

2.4 Modeling Transit Flow

In this section we will discuss the results of the primal min-cost flow optimization (2) and compare the results from the modifications discussed in section 1.2. In nearly all the model variant tested, the segment of 4,5 train going from 86th Street to 59th Street had the highest flow, followed by the adjacent sections below to 42nd Street and above to 125 St. These are widely reported to be the most overcrowded subways in New York City, and have


🟦 $500-$750 per bed per month
🟩 $750-$950 per bed per month
$950-$1200 per bed per month
$1200-$1450 per bed per month
🟥 $1450+ per bed per month

A heat-map generated from the per-station prices **Φ** inferred by the optimization in (3)

A map of rent costs in NYC provided by curbed magazine

Figure 3: Here we can see that the pricing model works well enough to identify Lower Manhattan and parts of Brooklyn as valuable areas, however

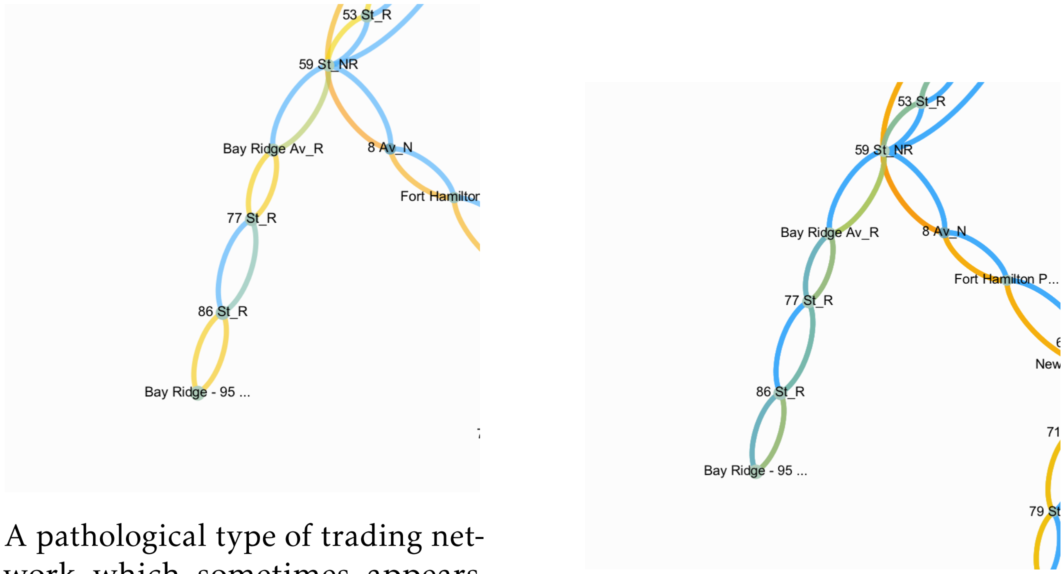
been the focus of relief efforts in the form of the planned Second Avenue line. These high-traffic track segments are visible in dark red in Figure 5. In fact, according to these results, about 13 per-cent of all daily riders pass through this segment of track every day (nearly 800000 people).

Modified Model

A primary observation that led us to design additional constraints for this problem was that the original trading model will necessarily use half or less of the available arcs for transport since each track is two-way. In figure 9, we can see that the entrance - exit flow equivalency constraint does lead to a more realistic distribution of flows.

Entrance - exit flow equivalency does lead to some unnatural artifacts, such as isolated two-station commuter relationships shown below:

One main reason to have pause with our preliminary results is that the network structure is a work in progress as of writing. Although all stations are present, and most transfers are available, there are issues that are unaccounted for. For example, there is not a penalty to prevent heavily used stations from “sending commuters to themselves”. Eventually we must include ”reverses” as a type of penalized transfer, if we hope for the



A pathological type of trading network which sometimes appears when using constraint (6)

The expected behavior

Figure 4: Here we can see that the pricing model works well enough to identify Lower Manhattan and parts of Brooklyn as valuable areas, however

entrance/exit lower bounds to be of help.

3 Evaluating Transit Proposals

Lastly, we will highlight the utility of optimal transport models by evaluating two proposed transit plans. We use the optimal flow framework to specifically evaluate the effect of these plans on *current riders*, not new ones that might be attracted by the new system. Correspondingly, we only include stations which purport a subway transfer capability, and make no assumptions about the entrances and exits of passengers on newly planned, nonexistent stations. As it were, we only modify our model through the addition of edges, and not nodes. These edges are equipped with scalar costs determined based on the total trip times advertised in the reports [3][1]: 86 minutes end to end trip time for BQX, and 96 minutes total for Triboro (implied by their average speed). These time allotments were partitioned according to rough distance measurements. In tables 1 and 2, we can see that the Triboro plan performs the best in 3 of four matchups. The units in table one, can be interpreted as man-minutes and the differential between BQX and Triboro is equivalent to roughly 3.37 years of wasted New-Yorker

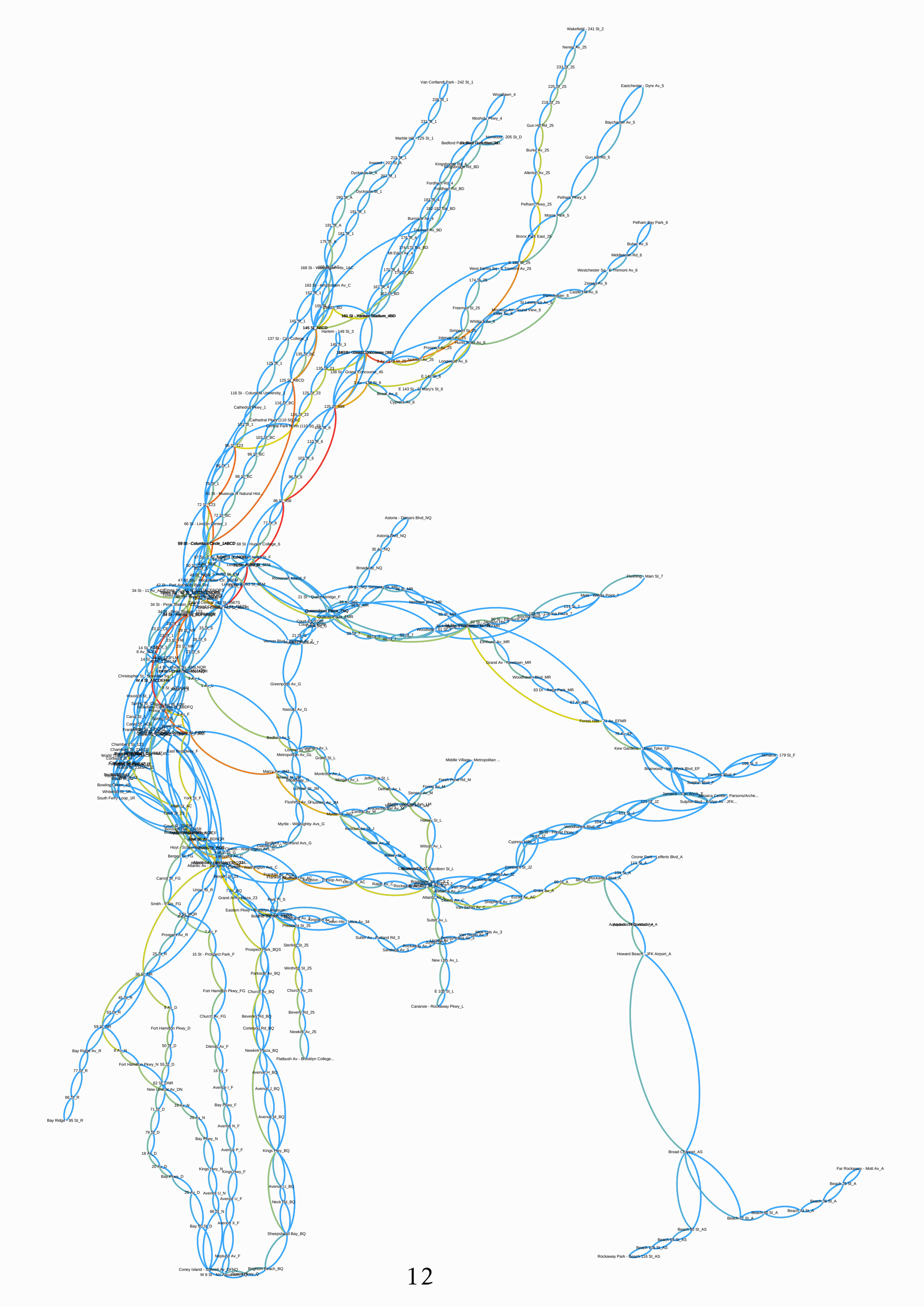


Figure 5: Commuter flows over the NYC subway system, using an unmodified min-cost flow solver, rendered in gephi

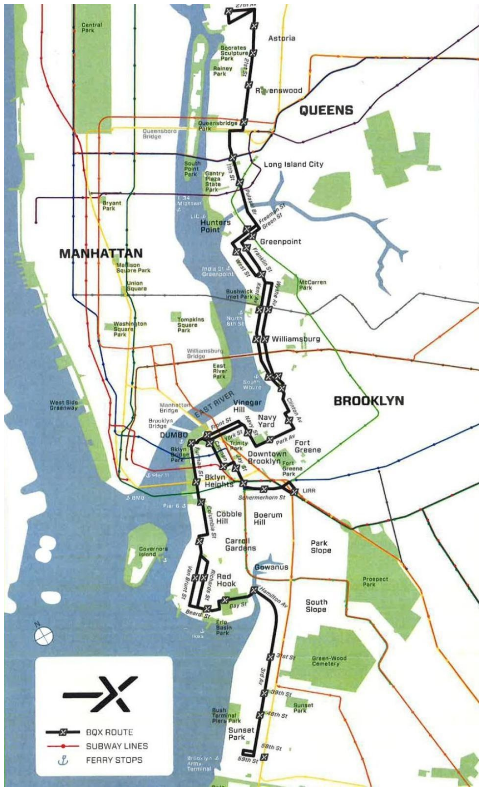


Figure 6: The BQX proposal for a waterfront streetcar.

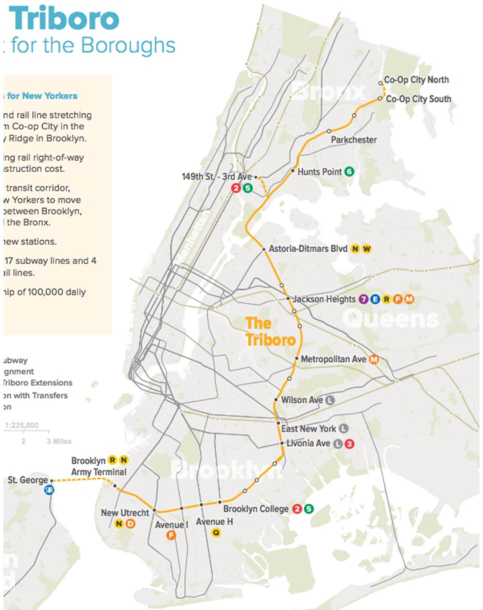


Figure 7: The Triboro proposal

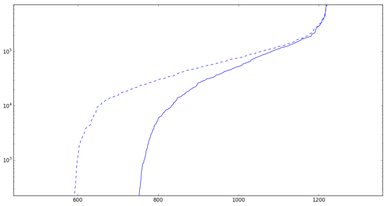


Figure 8: Ordered flow levels: Dotted line is using constraint (6), and solid line is an unmodified netowrk.

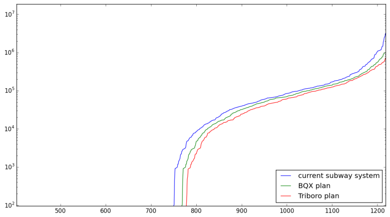


Figure 9: Ordered cost-flow levels: With the addition of both systems, our system would become more efficient overall.

time... per day.

Table 1: Min Cost Flow, $\mathbf{\Pi} \cdot \mathbf{c}$

	Trading Model	**Our Model** (using constraint (6))
Current Subway	94407145.75	108767423.946
BQX	93499050.35	108292983.735
Triboro	**91729580.7211**	**106099950.036**

Table 2: Maximum flow level (proxy for congestion) (**4,5** 86th - 59th st), $max(\mathbf{\Pi})$

	Trading Model	**Our Model** (using constraint (6))
Current Subway	726006.117825	786578.750725
BQX	727070.222584	794036.857998
Triboro	726006.117825	**723777.285966**

4 Conclusion

In this paper we began to explore the possibilities for applying optimal transport techniques to subway ridership data and discussed avenues for customization constraints for this setting. We also covered some particulars about what was necessary to extract said data necessary. Moving forward, there is room for adding temporal temporal analyses, as the turnstile data is updated weekly Please. There is definitely a place for the *swipe* data `http://web.mta.info/developers/fare.html` in a subway usage model, because it provides useful per-station demographic data updated on a weekly basis. Follow the progress of this project on `https://github.com/PorkShoulderHolder/transit`.

References

[1] R. P. Association. The triboro: Transit for the boroughs, apr 2016.

[2] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks, 2009.

[3] N. DOT. Brooklyn queens connector rapid assessment, 2016.

[4] G. T. Esteban G. Tabak. Data-driven optimal transport. Manuscript, 2014.

[5] A. Galichon. Optimal transport methods in economics. Preprint, nov 2015.

[6] W. McKinney. pandas: a foundational python library for data analysis and statistics.

[7] W. McKinney. Data structures for statistical computing in python. In S. van der Walt and J. Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[8] F. Santambrogio*. Models and applications of optimal transport theory. Grenoble, 2009.

[9] S. E. F. William W. Cohen, Pradeep Ravikumar. A comparison of string distance metrics for name-matching tasks. *American Association for Artificial Intelligence*, 2003.

[10] Q. Xia. An application of optimal transport paths to urban transport networks. *DISCRETE AND CONTINUOUS DYNAMICAL SYSTEMS*, 2015.