The George Washington University
Columbia College of Arts and Science
Department of Statistic
Section 6289

Final Report—Network Flows

Student Name and GWid
Jiaying Liu G45268292

Finished Date
DEC/10/2018

Contents

1 Introduction: Traffic Matrix Estimation **1**

2 Method of Traffic Matrix Estimation **1**

2.1 Notation 1

2.2 The Tomogravity Method 1

3 Background **2**

4 Traffic Matrix Estimated **3**

5 Appendix **8**

5.1 Whole dataset for predicr value 8

5.2 Code 12

1 Introduction: Traffic Matrix Estimation

Many networks serve as conduits—either literally or figuratively for flows, in the sense that they facilitate the movement of something, such as materials, people, or information.

Flows are at the heart of the form and function of many networks, and understanding their behavior is often a goal of primary interest. Much of the quantitative work in the literature on flows is concerned with various types of problems involving.Questions that are of a more statistical nature typically involve the modeling and prediction of network flow volumes, based on relevant data, and it is upon these topics that we will focus here

In this project, rather than distinguished by whether the traffic matrix Z is observed or to be predicted, we will focus on the question of traffic matrix estimation. Assume that we are given sufficient information on marginal flow volumes (i.e., in the form of total volumes through vertices or over links) and on the routing of flows between origins and destinations, it is often possible to generate accurate predictions of Z using statistical methods.

Measurements of flow volumes on network links, in conjunction with knowledge of the manner in which traffic flows over these links, between origins and destinations, can be sufficient to allow us to accurately predict origin-destination volumes. This is known as the traffic matrix estimation problem.

2 Method of Traffic Matrix Estimation

2.1 Notation

let X_e denote the total flow over a given link $e \in E$, and let $X = (Xe)_{e \in E}$. The link totals in X can be related to the origin-destination flow volumes in Z through the expression $X = BZ$, where Z now represents our traffic matrix written as a vector and B is the so-called routing matrix, describing the manner in which traffic moves throughout the network.

The traffic matrix estimation problem then refers to predicting the Zi j from the observed link counts $X = (Xe)_{e \in E}$

2.2 The Tomogravity Method

In the tomogravity method, we use a penalized least-squares criterion of the form

$$(x - B\mu)^T(x - B\mu) + \lambda D(\mu||\mu^{(0)})$$

where again the goal is to minimize the overall objective function. Here the penalty

$$D(\mu||\mu^{(0)}) = \sum_{ij} \frac{\mu_{ij}}{\mu_{++}} log \frac{\mu_{ij}}{\mu_{ij}^{(0)}}$$

is the relative entropy 'distance' between a candidate solution μ and some pre-specified vector $\mu^{(0)}$. This quantity is also known as the Kullback–Liebler divergence and summarizes how similar the two "distributions" μ_{ij}/μ_{++} and $\mu_{ij}/\mu_{ij}^{(0)}$ are to each other, where μ_{++} and $\mu_{ij}^{(0)}$ are the sum of the elements in and $\mu^{(0)}$, respectively. It is always non-negative and will be equal to zero if and only if $\mu = \mu^{(0)}$

The notion of a gravity model enters into the tomogravity method by specifying $\mu^{(0)}$ to have a certain multiplicative form reminiscent of a simple gravity model. Specifically, we let

$$\mu_{ij}^{(0)} = Z_{i+}^{(0)} Z_{+j}^{(0)}$$

be the product of the net out-flow and in-flow at vertices i and j, respectively. In addition, the constraint $\mu_{++} = Z_{++}^{(0)}$ is enforced, where $Z_{++}^{(0)}$ is the total traffic in the network. Importantly, these values all can be obtained through appropriate summation of the elements in the vector x of observed link counts.

Also necessary to completely specify is the value λ. This value acts as a smoothing parameter,with larger values encouraging solutions that are closer to the pre-specified $\mu^{(0)}$. A value of $\lambda = (0.01)^2$ is report to work well in practice.

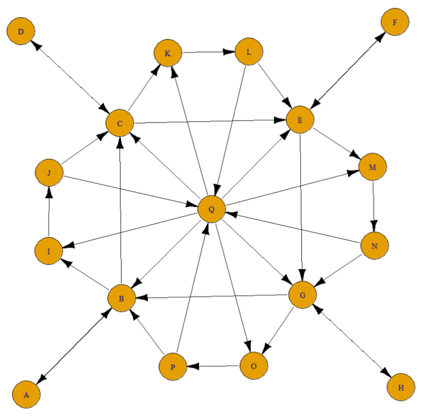
The package networkTomography implements the tomogravity method, with IPFP adjustment, in the function tomogravity. We illustrate by applying it to city transportation.

3 Background

In this example, we have two dataset and one map of cities transportation with

- There are 17 cities in the map with directed edges (one-way road).
- The cities A,D,F and H were isolated for a long time but about ten years ago, in each of the cities A, D, F and H, a company built a site and a two-way road to connect the site to the nearest city (B, C, E and G, respectively).
- Each road (one-way or two-way) is 100 miles long (and the edges in the graph are not proportional to this road distance).

Figure 1: Map of 17 cities



- After the road constructions, the company transported cargoes among the cities A, D, F and H. And the the shortest paths were used by the company for the transportations among the cities A, D, F and H.
- Eight years later, the company performed a review for the purpose of transportation efficiency improvements. However, the data (amounts of cargoes) transported among the cities A, D, F and H could not be found at the time of review. Instead, other related data were found: the amounts of cargoes transported through different roads recorded in each month of the previous eight years
- Dataset " FinalProjectRoad" which provide the amounts of cargoes transported through different road
- Dataset "FinalProjectTransportation" which provide the cargoes transported among the cities

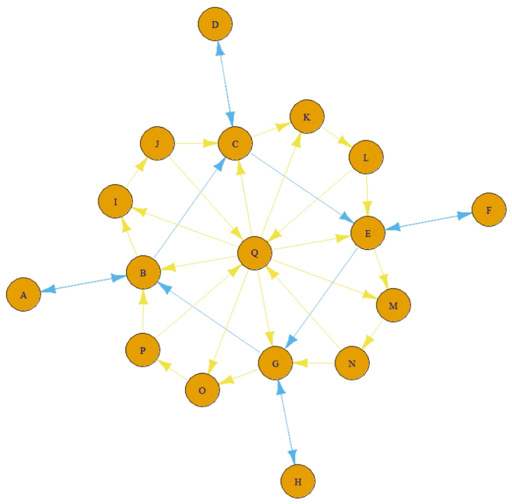
4 Traffic Matrix Estimated

With the information of monthly cargoes transported through different road in previous eight years—Link count X, we want to estimate the traffic matrix Z.
There are three important matrix we need to specify.

- X Link flow volumes Matrix which is the matrix of mouth vs. link
- B Routing Matrix which is the matrix of link vs.origin-destination
- Z Traffic matrix which is the matrix of month vs. origin-destination

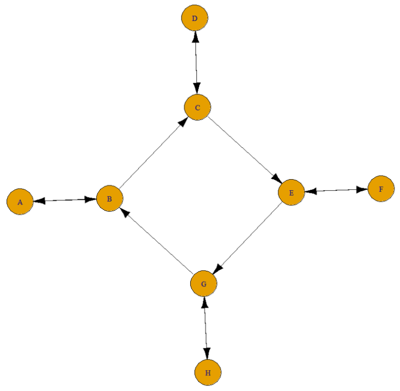
From the "FinalProjectRoad" dataset, we can find that there are some roads with no cargo transported.So we set the edges which have cargoes transported as blue and edges with zero cargo transported as yellow, than we can get the network like the following figure.

Figure 2: Edge with and without cargo transporting



Road without cargoes means that the company do not need those road to transport and it also mean that the shortest path do not contains those road.So we can simplify the dataset, deleting all the columns without any cargoes transported.Then the above figure can be simplified like figure 3.

Figure 3: Edge with and without cargo transporting



From the background, we know that each road is 100 miles long and the shortest paths were used by company for the transportation among the cities A,D,F and H.

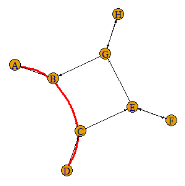
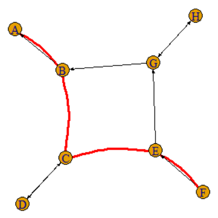
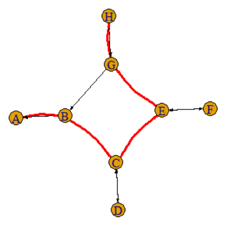
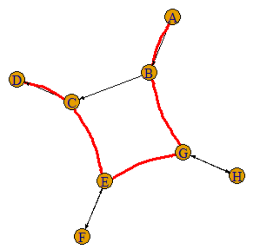
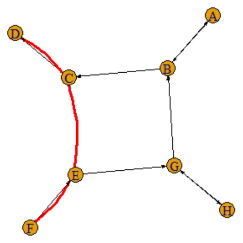
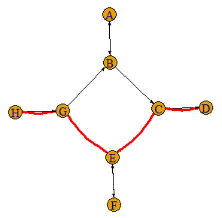
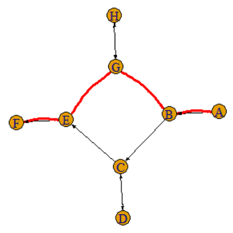
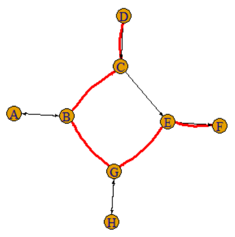
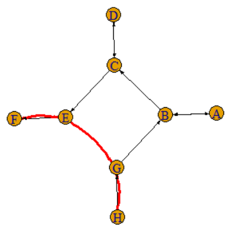
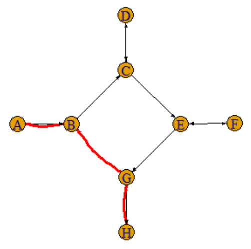
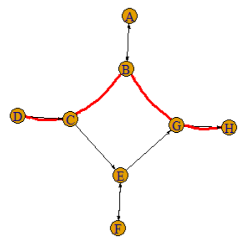
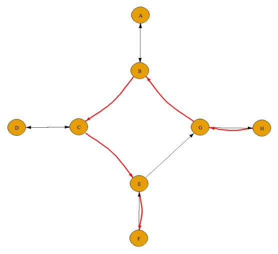
So the next step we need to find the shortest paths for each origin and destination in the figure 3.

In the table 1, it show all the shortest paths for each pair of destination. Based on this shortest paths, we can create a routing matrix with rows corresponding to links and columns corresponding to pairs of vertices. Note that the order of row item and column item should follow by the two dataset.

	A-D	A-F	A-H	D-A	D-F	D-H	F-A	F-D	F-H	H-A	H-D	H-F
A-B	1	1	1	0	0	0	0	0	0	0	0	0
B-A	0	0	0	1	0	0	1	0	0	1	0	0
B-C	1	1	1	0	0	0	0	1	0	0	1	1
C-D	1	0	0	0	0	0	0	1	0	0	1	0
C-E	0	1	1	1	1	1	0	0	0	0	0	1
D-C	0	0	0	1	1	1	0	0	0	0	0	0
E-F	0	1	0	0	1	0	0	0	0	0	0	1
E-G	0	0	1	1	0	1	1	1	1	0	0	0
F-E	0	0	0	0	0	0	1	1	1	0	0	0
G-B	0	0	0	1	0	0	1	1	0	1	1	1
G-H	0	0	1	0	0	1	0	0	1	0	0	0
H-G	0	0	0	0	0	0	0	0	0	1	1	1

After constructing the routing matrix B, we are going to construct the matrix of link flow

Table 1: Shortest Paths for Each Pair

A-D	A-F	A-H
A-B-C-D	A-B-C-E-F	A-B-C-E-G-H
		
D-A	D-F	D-H
D-C-E-G-B	D-C-E-F	D-C-E-G-H
		
F-A	F-D	F-H
F-E-G-B-A	F-E-G-B-C-D	F-E-G-H
		
H-A	H-D	H-F
H-G-B-A	H-G-B-C-D	H-G-B-C-E-F
		

volumes matrix X which are happen to the "FinalProjectRoad" dataset. We delete the zero column and keep the column with data and we find that there are 12 col and 96 rows. And this is our X

With the "FinalProjectTransportation" dataset, we are able to get the actual traffic matrix which can be used to estimate the predicted traffic matrix made by the tomogravity method.

After all this preparation, we are ready to do the estimation with the statement TOMOGRAVITY. Because the insensitive of λ for the algorithm, it is easy to choose an appropriate λ for the penalty. So in this place, wo choose a vaue from the middle of the insensitive range $\lambda = 0.01$

```
tomo.fit=tomogravity(x_1,b_1,0.01)
zhat=tomo.fit$Xhat
```

With this two statement, we are able to create the prediction of traffic matrix for all pair of destination and 8 years data. The summary of prediction will show in the table 3, the complete result will put into appendix.

Table 2: Simplify prediction for traffic matrix

Month	A-D	A-F	A-H	D-A	D-F	D-H
1	2559	31476	6479	21763	31644	6514
2	961	18739	4466	41781	7711	183
3	1017	19193	5798	39142	9880	2985
4	573	4010	11008	158100	10626	29178
5	0	0	0	50469	0	2241
6	561	6338	3153	42548	3872	1926
93	3023	18792	11788	13317	21992	13795
94	4856	21370	12334	12938	21435	12372
95	3540	19017	12247	13799	22402	14426
96	3686	20460	15706	16596	25258	19387
Month	F-A	F-D	F-H	H-A	H-D	H-F
1	75132	5331	22484	4136	294	3612
2	777993	11221	34214	6080	88	1711
3	471813	8205	35984	5493	96	1802
4	797809	14128	147248	5366	95	665
5	849918	17325	37728	3910	80	0
6	744927	12272	33727	4831	80	899
93	12987	3465	13452	2942	785	4880
94	12292	4111	11754	3350	1121	4930
95	13184	4013	13782	3466	1055	5667
96	13129	3441	15335	2798	733	4071

With the prediction, we can use it to compare with the actually traffic matrix.

Figure 4: Compare predict with actually data

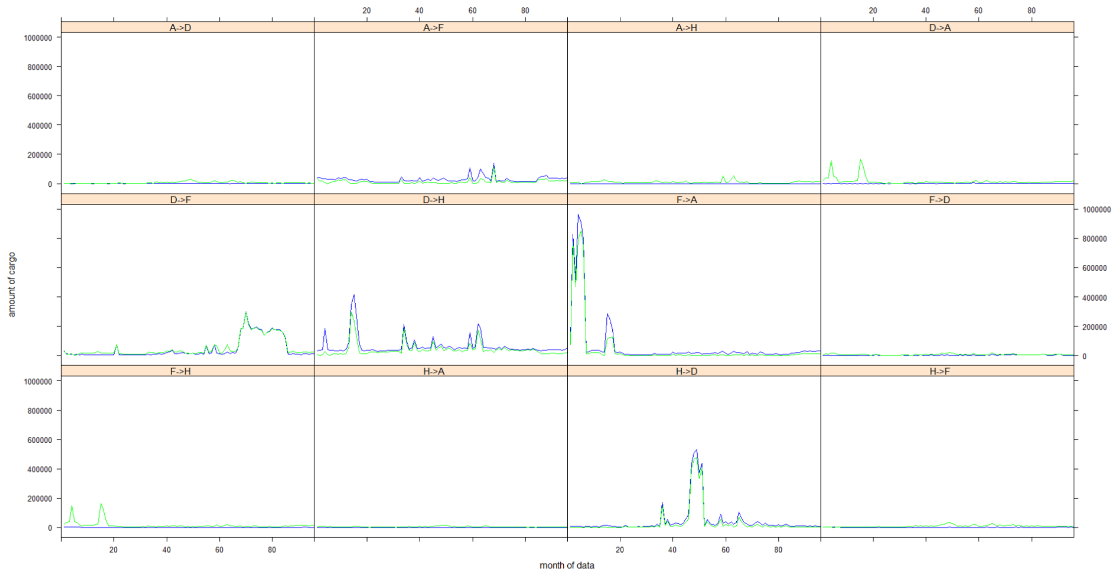


Figure 5: residual of prediction vs Z

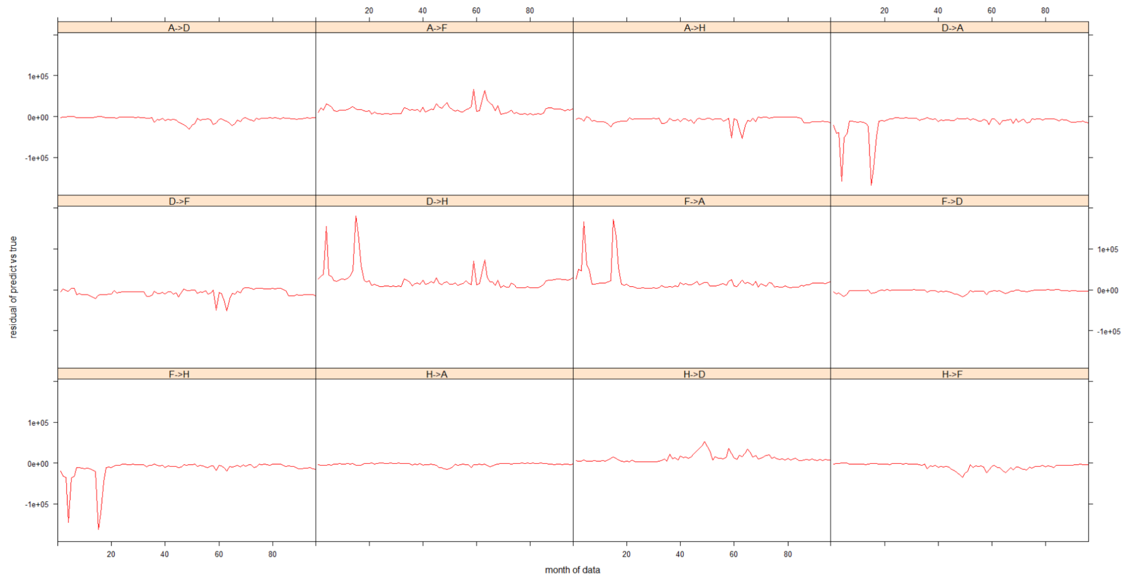


Figure 4 show the result of predict value compared with the true value, the green line is the predict value and the blue one is the true value. We can see the predict value is approximate to the true value. It is fit well. And the figure 5 show the residual of prediction vs true value which can show the result more directly. We can see there are still some place exist poor prediction like in D-H, F-A and F-H.

5 Appendix

5.1 Whole dataset for predicr value

Table 3: Complete dataset for predict traffic matrix

Month	**A-D**	**A-F**	**A-H**	**D-A**	**D-F**	**D-H**
1	2558.5	31475.5	6478.8	21763.2	31643.6	6514.5
2	960.7	18739.4	4465.5	41780.6	7711.1	1837.7
3	1017.2	19193.2	5797.8	39141.9	9879.7	2985.2
4	572.9	4009.9	11008.3	158100.0	10625.9	29177.5
5	0.0	0.0	0.0	50468.7	0.0	2240.7
6	561.0	6338.5	3152.9	42547.5	3872.0	1926.2
7	3073.8	14954.0	11395.2	11080.8	15176.4	11564.6
8	3090.0	14412.8	9659.2	11175.4	17389.7	11653.5
9	2482.0	25733.5	12267.0	13138.3	16875.0	8043.9
10	2610.0	19687.0	12160.0	14386.1	16810.5	10382.2
11	3247.2	25520.0	12247.5	13147.8	17054.8	8185.7
12	2497.3	25129.9	14257.0	14429.9	17694.6	10038.5
13	3090.8	6512.2	17903.8	16862.3	20893.5	57442.6
14	1346.9	2201.6	25442.4	23462.9	25855.0	298812.9
15	619.7	1167.7	16852.3	167899.1	16378.0	236419.2
16	718.8	1875.0	14739.7	126398.3	16307.9	128202.7
17	1524.4	10665.1	12636.1	48123.8	15313.4	18142.0
18	2437.7	14133.6	11617.3	11588.8	15334.6	12604.7
19	2290.8	12998.3	10612.4	9413.3	14141.6	11545.3
20	2399.2	14803.4	11396.7	11368.0	15208.8	11710.2
21	2923.1	9716.0	3014.6	8095.9	76280.2	23666.7
22	4522.0	3559.9	7694.2	5696.0	11643.9	25161.6
23	1721.5	2731.9	6046.6	5520.2	10884.2	24088.8
24	1662.2	2261.5	5038.0	3045.1	9356.8	20846.1
25	1529.0	1834.5	4994.7	3431.6	8467.2	23052.1
26	1672.9	2306.3	5805.0	4028.3	9406.3	23673.5
27	1915.7	1838.0	5195.5	3736.2	8644.5	24437.9
28	1622.7	1669.7	4973.4	3367.8	8533.6	25416.1
29	2195.1	1808.4	5190.9	3669.1	9553.7	27424.7
30	2041.1	1487.9	4673.7	3860.6	8864.9	27843.5
31	2452.7	1948.1	5160.4	3940.5	10030.1	26571.0
32	2793.8	1642.9	4383.8	3974.0	9670.6	25807.5
33	4476.8	23981.6	16788.0	9543.9	23913.7	16746.5
34	4454.4	1968.2	17075.3	6480.2	21719.6	188391.8
35	2531.3	3694.8	13650.1	6395.8	20278.8	74929.6
36	13834.1	1362.8	4406.6	3248.0	10759.9	34791.1
37	7024.7	5251.8	9519.7	5360.9	20348.9	36884.6
38	9064.2	2016.0	9493.8	6620.9	19223.4	90516.4
39	4954.1	4052.3	7085.3	6317.0	19718.2	34479.5
40	10396.3	17394.5	12165.5	12644.4	32397.5	22658.9
41	6876.5	3433.3	5402.2	8175.2	28036.3	44118.3

42	8083.2	6525.4	5663.8	9945.6	37759.9	32774.2
43	7497.3	9086.8	11783.6	8796.8	24815.7	32178.4
44	9174.7	5523.5	8759.2	7939.0	21882.8	34699.3
45	14738.6	5664.2	17644.2	10498.3	31819.3	99123.6
46	17319.6	5402.0	7002.5	9918.2	26300.4	34095.8
47	19631.2	1161.9	3722.9	4935.0	15833.3	50725.0
48	25562.1	1555.9	6370.6	5956.5	14913.1	61042.6
49	31144.8	2325.4	6482.0	5575.6	12492.9	34825.2
50	21584.8	1622.5	4383.8	6134.6	14301.0	38645.7
51	17342.4	841.0	4228.4	3933.7	10054.5	50562.7
52	4371.2	4274.1	8978.4	8287.1	18412.8	38679.4
53	10673.0	3186.2	5040.8	5411.2	19131.3	30267.2
54	6457.3	2723.4	6070.2	7052.1	14809.3	33016.4
55	6675.2	8916.7	5054.1	10938.7	70028.6	39684.6
56	6168.6	9790.3	10913.6	11342.8	21513.9	23978.5
57	9194.9	6857.0	9026.1	7807.9	26143.2	34415.9
58	20328.1	9130.8	4162.1	14253.0	75767.2	34535.6
59	15125.1	39663.7	52068.2	19798.7	66721.2	87594.4
60	7821.2	2702.8	6060.6	6442.0	16502.9	37013.8
61	5887.8	3330.6	8860.0	6359.6	16982.1	45169.6
62	10646.0	6954.8	29345.2	14121.5	41269.8	174168.0
63	10823.7	35200.6	53750.7	20502.8	75154.0	114766.0
64	17572.8	29937.7	22034.3	10314.1	32418.3	23860.0
65	23489.1	8063.8	9872.7	9465.1	28285.9	34635.4
66	17416.0	9696.4	11065.9	8963.3	27881.3	31815.6
67	9046.4	5701.1	4805.5	6874.3	45851.7	38635.3
68	12932.3	114251.9	13300.2	17296.0	183964.5	21414.7
69	4129.3	7436.3	1513.8	6419.3	186023.1	37869.8
70	3269.9	16115.0	2668.9	14609.6	293381.6	48584.1
71	6969.2	8466.3	1425.4	8898.3	217750.8	36664.3
72	9003.4	6176.7	1982.6	8270.8	177561.0	56984.8
73	11119.3	20348.6	4447.1	15184.5	183805.9	40158.9
74	4196.5	13827.9	2223.1	14919.4	193781.0	31154.5
75	7596.3	6933.5	1527.3	5962.7	176108.9	38782.8
76	4044.2	8445.2	1659.6	7661.0	173670.3	34131.8
77	4193.1	7840.9	1992.2	5607.8	135741.6	34486.4
78	4744.1	8292.1	1675.9	7300.5	156078.7	31546.2
79	2562.3	9656.2	1858.5	8310.5	160739.0	30939.1
80	4900.9	8073.0	1554.9	6942.6	184847.1	35606.8
81	2843.2	8587.7	1839.2	5405.2	174950.4	37473.3
82	4090.3	7961.9	1565.2	6744.5	173926.1	34195.0
83	5375.6	6242.4	1530.3	5437.6	173485.7	42528.6
84	3186.8	9992.1	2154.0	8267.9	156555.8	33749.2
85	4351.6	26784.2	4738.2	10811.1	128939.6	22812.5
86	4942.9	30617.4	15433.4	8576.7	21336.1	10754.4
87	5568.3	29335.0	15078.3	8857.3	22391.9	11509.5
88	5418.2	33945.9	15738.8	9963.0	28682.5	13299.4
89	6596.6	17241.5	12244.4	12516.7	18453.3	13104.6

90	6073.0	19408.6	13259.7	14704.6	21378.6	14606.7
91	5205.9	17758.6	13299.0	13594.5	19594.5	14674.5
92	4666.5	21038.9	12261.5	13742.9	22608.0	13176.3
93	3023.2	18791.7	11788.3	13317.4	21992.2	13795.0
94	4856.5	21370.2	12334.1	12937.7	21435.2	12372.2
95	3540.0	19017.5	12246.7	13799.2	22402.0	14426.2
96	3685.6	20460.0	15706.5	16596.5	25257.8	19387.2

Table 4: Complete dataset for predict traffic matrix

Month	**F-A**	**F-D**	**F-H**	**H-A**	**H-D**	**H-F**
1	75131.9	5330.8	22484.3	4136.4	293.9	3612.3
2	777992.7	11220.7	34213.7	6080.3	87.7	1710.6
3	471813.0	8204.6	35983.6	5492.9	95.5	1801.9
4	797809.2	14127.9	147247.9	5365.7	95.0	664.9
5	849918.2	17325.2	37727.6	3909.5	79.7	0.0
6	744926.9	12271.9	33726.8	4831.2	79.6	898.9
7	12011.7	3793.2	12536.7	2158.3	681.6	3315.6
8	10402.4	3252.8	10847.1	2226.3	696.1	3247.1
9	21332.6	2558.4	13061.7	1851.9	222.0	2302.1
10	19752.9	3052.3	14254.7	2165.0	334.6	2524.3
11	20947.6	3298.5	13041.7	2508.0	394.8	3102.7
12	20698.6	2571.0	14401.7	1866.8	231.8	2332.5
13	4753.6	3427.9	16193.5	2279.3	1643.9	3463.4
14	1605.4	2207.5	20445.5	1011.8	1391.4	2274.3
15	115084.5	9421.9	162026.1	5115.2	418.7	789.2
16	120727.4	8872.5	122426.8	5581.6	410.2	1070.1
17	124339.1	6565.2	46878.3	4992.1	263.6	1844.2
18	10508.4	2586.5	11429.1	1697.7	417.8	2422.9
19	7512.8	2183.0	9214.9	1362.4	395.7	2245.4
20	10890.8	2532.8	11218.1	1639.3	381.3	2352.7
21	2601.7	8171.5	7606.3	319.4	1003.0	3334.0
22	1271.1	3240.3	5616.5	2305.2	5876.1	4626.9
23	1218.6	1820.2	5317.3	785.7	1173.7	1862.7
24	434.1	1233.9	2971.8	458.5	1303.3	1773.3
25	492.7	1287.3	3308.9	509.2	1330.4	1596.4
26	666.8	1443.4	3918.2	591.0	1279.5	1764.7
27	554.5	1551.8	3627.0	757.0	2118.6	2032.3
28	428.5	1316.4	3234.3	535.8	1646.1	1694.4
29	478.6	1702.6	3577.4	788.2	2803.9	2309.7
30	525.6	1817.8	3791.3	865.8	2994.3	2182.8
31	665.3	4132.8	4485.5	401.6	2494.5	1981.2
32	605.6	2370.3	3932.9	1287.3	5037.8	2962.9
33	5310.8	3012.4	9318.9	1471.6	834.5	4471.2
34	209.3	2063.8	6084.9	1184.3	11676.5	5160.0
35	507.5	1584.5	5946.3	673.2	2101.6	3067.5
36	294.6	7191.2	3155.7	6080.7	148444.9	14622.7

37	770.5	3723.1	5301.9	2094.8	10122.3	7568.3
38	539.9	6014.6	7380.4	3609.7	40224.5	8949.9
39	1116.1	3788.1	6092.5	1958.7	6645.5	5435.3
40	6569.9	7796.2	11773.2	5645.5	6700.2	11209.4
41	1407.9	5860.5	7601.2	3875.0	16130.0	8054.3
42	2739.8	7416.6	9028.9	4446.0	12034.4	9715.8
43	2285.5	4811.4	8361.0	3127.8	6584.5	7979.7
44	1750.3	6138.2	7648.9	4646.2	16293.7	9810.7
45	1126.2	7360.5	10633.1	5992.1	39164.8	15051.9
46	2696.0	14481.2	9273.3	11017.1	59164.2	18446.0
47	349.9	13281.6	3598.4	10528.1	399449.7	23638.4
48	507.3	17384.9	5194.2	13602.0	466557.2	28398.0
49	660.1	19871.5	4121.7	15860.3	477507.1	35663.1
50	728.5	17006.5	4588.4	14162.6	330582.9	24846.3
51	252.7	11163.2	3247.2	9203.8	406723.1	19720.4
52	1696.2	3814.8	7918.0	2138.8	4809.5	4702.7
53	939.9	7222.6	5257.2	4895.0	37614.9	11231.0
54	1498.2	5780.0	7014.8	4220.3	16282.8	6867.1
55	2568.5	7030.7	9319.1	2462.1	6739.5	9002.4
56	5206.4	5801.2	11008.4	3897.4	4343.5	6893.6
57	1684.4	5944.4	7421.9	3718.3	13121.9	9786.3
58	3084.7	14547.9	7473.9	11131.2	52497.2	23580.9
59	4193.0	5989.3	18554.7	4385.3	6264.5	16428.9
60	982.7	5576.8	5647.7	4201.6	23842.5	8238.8
61	883.5	3779.2	6273.0	2547.0	10900.3	6166.9
62	1041.4	5077.4	12845.9	3652.0	17805.1	11630.7
63	3489.3	5236.2	19532.5	2875.5	4315.2	14032.5
64	3940.7	7262.6	9117.2	5750.1	10597.1	18052.3
65	2577.7	18068.6	9433.4	9970.8	69878.0	23990.7
66	2214.1	9989.8	7857.9	7153.9	32276.6	17972.0
67	1030.8	5562.2	5795.2	3445.9	18580.7	11715.2
68	8413.8	7244.3	10419.5	2173.0	1870.6	16526.0
69	579.8	5377.1	3420.5	551.2	5112.6	9207.7
70	2435.2	6726.4	8096.8	733.6	2026.3	9985.0
71	1152.9	8924.8	4750.9	1449.0	11217.3	13625.1
72	624.2	7168.9	4301.3	1947.6	22369.2	15345.0
73	4220.8	13319.1	11160.6	2838.5	8960.2	16396.1
74	4508.2	8105.0	9413.2	1658.5	2981.8	9824.7
75	448.6	5828.1	2918.0	1105.9	14366.6	13112.3
76	828.1	5100.1	3689.0	722.3	4447.6	9287.2
77	539.6	4168.6	3317.9	547.2	4228.1	7906.2
78	861.9	5769.3	3724.4	761.9	5100.0	8914.6
79	1438.2	4955.8	5354.2	477.8	1646.4	6204.6
80	608.3	5485.4	3119.7	666.6	6010.7	9901.7
81	463.1	4166.2	3210.0	248.3	2234.6	6749.2
82	582.2	4510.8	2951.3	582.2	4511.7	8782.2
83	356.0	4880.3	2785.0	663.5	9094.0	10556.9
84	1314.1	6653.8	5364.1	383.4	1941.4	6087.6

85	4072.9	6781.7	8595.1	576.1	959.3	5904.6
86	6010.1	2599.4	7535.8	1938.5	838.4	5193.4
87	6486.2	3097.7	8427.4	2299.1	1098.1	5784.0
88	6356.9	2901.8	8487.8	2029.5	926.2	5804.4
89	12235.9	8384.3	12810.3	3591.7	2461.6	6433.4
90	14468.4	8843.4	14371.9	3034.3	1854.5	5926.1
91	13141.6	7967.5	14185.0	2427.1	1471.3	5019.7
92	13765.5	5013.8	13198.7	3497.6	1274.1	5744.7
93	12986.6	3464.6	13452.4	2942.3	785.1	4880.3
94	12292.0	4111.3	11754.4	3350.4	1120.6	4929.5
95	13183.9	4013.2	13782.1	3465.5	1055.1	5666.8
96	13129.4	3440.8	15334.6	2797.7	733.3	4071.2

5.2 Code

```
library(networkTomography)
library(igraph)
library(networkTomography)
library(igraph)

edge_list=rbind(
edge_list=rbind(
  c("A","B"),
c("B","A"),
c("B","C"),
c("B","I"),
c("C","D"),
c("C","K"),
c("C","E"),
c("D","C"),
c("E","F"),
c("E","M"),
c("E","G"),
c("F","E"),
c("G","O"),
c("G","H"),
c("G","B"),
c("H","G"),
c("I","J"),
c("J","C"),
c("J","Q"),
c("K","L"),
c("L","E"),
c("L","Q"),
c("M","N"),
c("N","G"),
c("N","Q"),
c("O","P"),
c("P","B"),
c("P","Q"),
c("Q","B"),
c("Q","I"),
c("Q","C"),
c("Q","K"),
c("Q","E"),
c("Q","M"),
c("Q","G"),
c("Q","O")
)
c=graph_from_edgelist(edge_list)
plot(c,vertex.cex=8,vertex.dist=4,edge.arrow.size=0.25,
edge.color="black",main="Network of city Transportation ")


 c("A"),

c("B","C"),
c("B","I"),
c("C","K"),
c("C","E"),
c("D"),
c("E","M"),
c("E","G"),
c("F"),
c("G","O"),
c("G","B"),
c("H"),
c("I","J"),
c("J","C"),
c("J","Q"),
c("K","L"),
c("L","E"),
c("L","Q"),
c("M","N"),
c("N","G"),
c("N","Q"),
c("O","P"),
c("P","B"),
c("P","Q"),
c("Q","B"),
c("Q","I"),
c("Q","C"),
c("Q","K"),
c("Q","E"),
c("Q","M"),
c("Q","G"),
c("Q","O")
)
c_2=graph_from_edgelist(edge_list_2)
#delete selfloop
c_2=simplify(c_2,remove.loops = TRUE)
plot(c_2,vertex.cex=8,vertex.dist=4,edge.arrow.size=0.25,
edge.color="black",main="Network of original city Transportation ")


edge_1=c(1,1,1,2,1,2,1,1,1,2,1,1,2,1,1,1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2)
E(c)$trans=edge_1
plot(c,vertex.cex=8,vertex.dist=4,edge.arrow.size=0.25,
main="Network of city Transportation ",edge.color=edge_1*2)



edge_list_1=rbind(
  c("A","B"),
c("B","A"),
c("B","C"),
c("C","D"),
c("C","E"),
c("D","C"),
c("E","F"),
c("E","G"),
c("F","E"),
c("G","H"),
c("G","B"),
c("H","G"))
c_3=graph_from_edgelist(edge_list_1)
plot(c_3,vertex.cex=8,vertex.dist=4,edge.arrow.size=0.25,
edge.color="black",main="Network of original city Transportation ")



est_data=read.csv('C:/Users/jiayingliu/Downloads/FinalProjectRoad.csv',
header = TRUE)

#get the dataset about road tranport
x=est_data[,2:13]

#create adjance matrix for B
b=read.csv("C:/Users/jiayingliu/Desktop/final/B.csv",h=TRUE)
b=b[1:12,2:13]

#read in validation data
z=read.csv('C:/Users/jiayingliu/Desktop/final/FinalProject.csv',h=TRUE)
z=z[,2:13]
```

```
b_2=sapply(b,as.numeric)
b_2[b_2==0]=2
for (i in 1:12){
plot(c_3,vertex.cex=8,vertex.dist=4,edge.arrow.size=0.25,\\
main="Network␣of␣shortest␣path␣␣",edge.color=t(b_2[,i])*3)}



x_1=sapply(x,as.numeric)
b_1=sapply(b,as.numeric)


x_1=sapply(x,as.numeric)
b_1=sapply(b,as.numeric)
z_1=sapply(z,as.numeric)


tomo.fit=tomogravity(x_1,b_1,0.01)
zhat=tomo.fit$Xhat

#write.csv(data.frame(zhat),"C:/Users/jiayingliu/Desktop/final/zhat.csv")

nt=nrow(z_1)
nf=ncol(z_1)

t.dat=data.frame(z=as.vector(c(z_1)),
zhat=as.vector(c(zhat)),t=c(rep(as.vector(est_data$month),nf)))

od.names=c(
  rep("␣A−>D␣␣␣␣",nt),
  rep("␣A−>F␣␣␣␣",nt),
  rep("␣A−>H␣␣␣␣",nt),
  rep("␣D−>A␣␣␣␣",nt),
  rep("␣D−>F␣␣␣␣",nt),
  rep("␣D−>H␣␣␣␣",nt),
  rep("␣F−>A␣␣␣␣",nt),
  rep("␣F−>D␣␣␣␣",nt),
  rep("␣F−>H␣␣␣␣",nt),
  rep("␣H−>A␣␣␣␣",nt),
  rep("␣H−>D␣␣␣␣",nt),
  rep("␣H−>F␣␣␣␣",nt))

t.dat=transform(t.dat,OD=od.names)

xyplot(z~t | OD, data=t.dat,
panel=function(x, y, subscripts){
```

```
panel.xyplot(x, y, type="l", col.line="blue")
panel.xyplot(t.dat$t[subscripts],
t.dat$zhat[subscripts],
type="l", col.line="green")
}, as.table=T, subscripts=T, xlim=c(0, 96),
xlab="month of data", ylab="amount of cargo")

t.res=data.frame(z=as.vector(c(z_1)),zhat=as.vector(c(zhat)),
res=as.vector(c(z_1)-c(zhat)),t=c(rep(as.vector(est_data$month),nf)))
t.res=transform(t.res,OD=od.names)

xyplot(res~t | OD, data=t.res,
panel=function(x, y, subscripts){
panel.xyplot(x, y, type="l", col.line="red")
}, as.table=T, subscripts=T, xlim=c(0, 96),
xlab="month of data", ylab="residual of predict vs true")
```