STAT 6289 Network Final

Chenrui Xu

I Background

This is the final project of the STAT 6289 Network. The data we will use is based on the flow of transportation between 17 cities. We can name after these cities from A to Q and there are somehow some roads to get them connected. As we should know that all roads here are one-way (directed edges). In the beginning, four cities A, D, F, H are isolated as there were no roads connecting them. And later, two-way roads were build to transport cargo between cities. The flow of cargoes was recorded when they went through one link and also the number of cargoes transported among A, D, F, H (that is what is lost and we are going to estimate).

Some knowledge we will use in the model are:

OD matrix: That is one kind of data frame we will use. It will record where the flow starts (Origin) and where the flow ends (Destination).

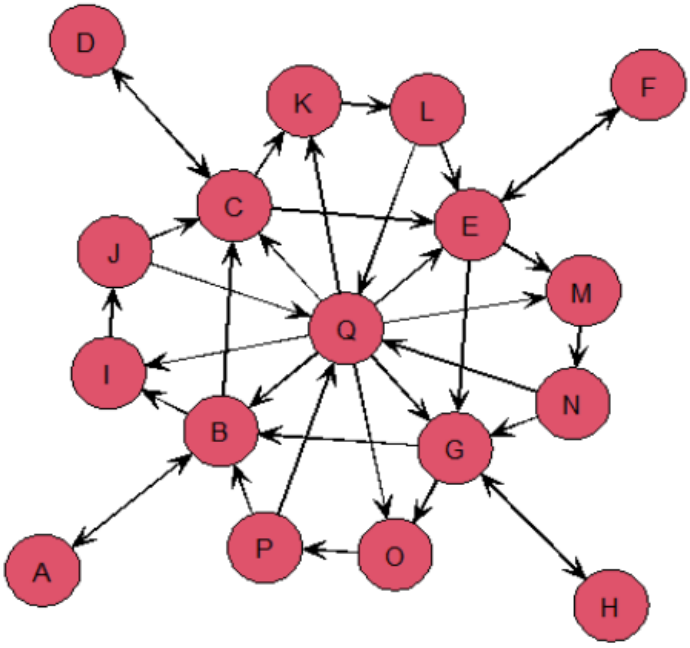
Gravity Model (tomo-gravity model): The regular gravity model will describe how the interaction between nodes. If we only know the amount of flow that goes through links and the timeline, we can use tomo-gravity model to predict the OD matrix. The model is based on the linear formation and the objective function is the loss function with the penalized item.

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

Here is the formula of the tomo-gravity model.

II Task I

In this part, we should build the tomo-gravity model and predict the amount of cargoes among points A, D, F, H.



Some analysis of the data processing:

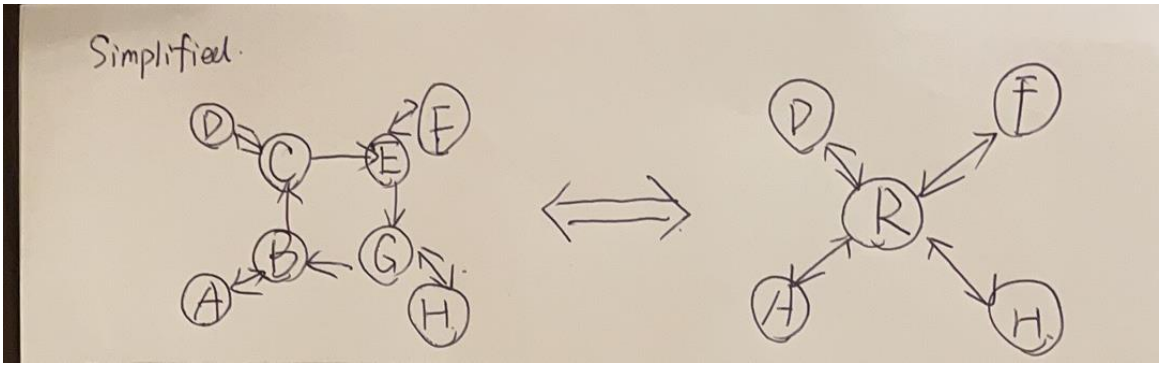
If we want to use the tomo-model, we need to know what are X, B, Z. Z is supplied but that is the OD matrix we need to estimate and compare. X is given as “Road” Matrix, which we will talk about later. B is the rounting matrix.

Next step: B matrix

This is the core step during the whole analysis.

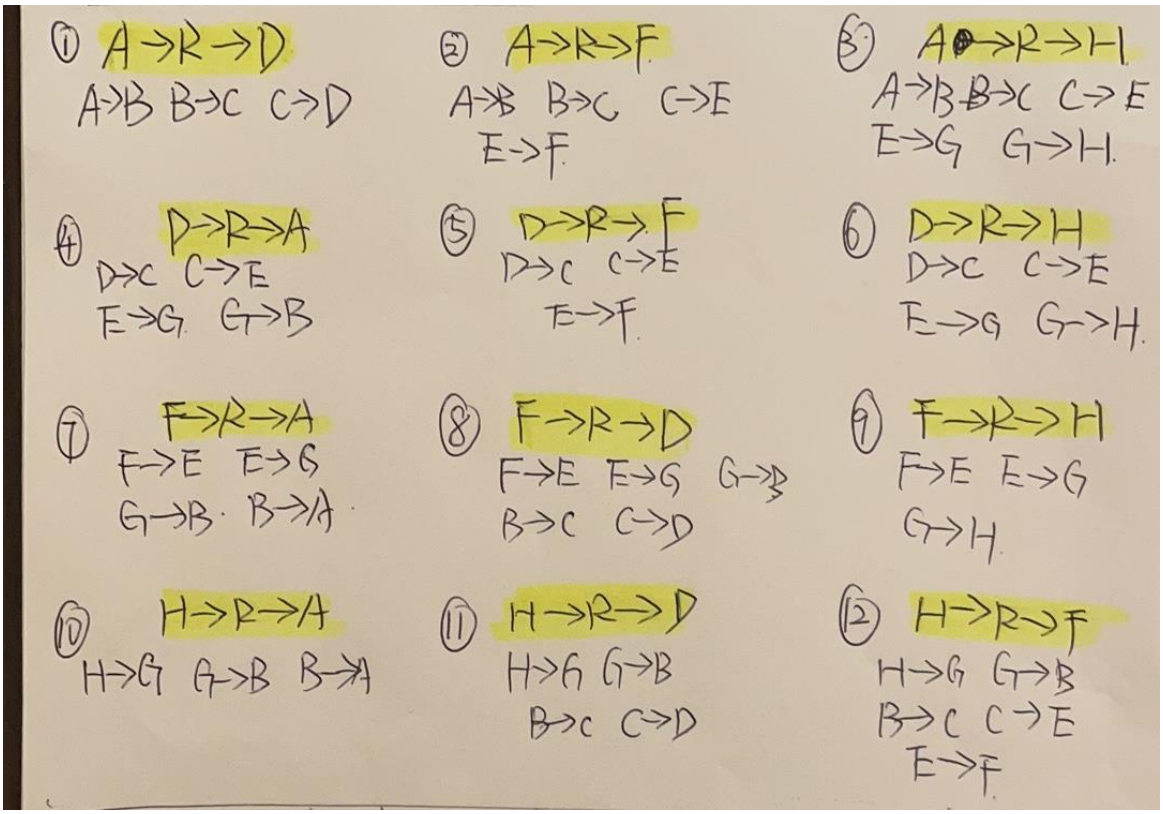
If we want to analysis the way among A, D, F, H, there are four points we have to go through if those ways are the shortest paths. They are B, C, E, G.

These four points can actually be seen as a system as a router same as the exapmle in the lecture notes. **So the whole 17 points network can be simplified into four points and a router system.**

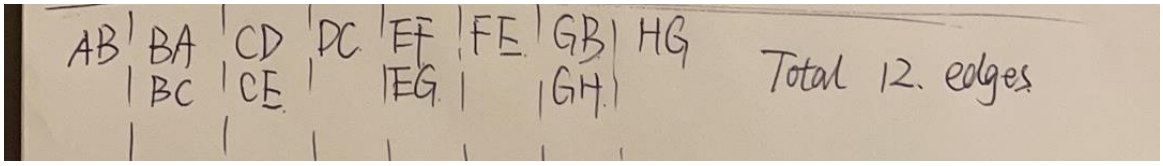


We do not need to know the data not related to those links, so next step, I will use graphs to show

how many links do we need to use here.



So, there are 12 OD links here (Origin to Destination). Their corresponding composed edges are showing below. So we can know which are useful.



After artificial counting, it is not hard for us to find that there are totally 12 edges will be used into the analysis.

Based on that, we can easily get our B matrix.

	A	B	C	D	E	F	G	H	I	J	K	L
1	1	1	1	0	0	0	0	0	0	0	0	0
2	0	0	0	1	0	0	1	0	0	1	0	0
3	1	1	1	0	0	0	0	1	0	0	1	1
4	1	0	0	0	0	0	0	1	0	0	1	0
5	0	1	1	1	1	1	0	0	0	0	0	1
6	0	0	0	1	1	1	0	0	0	0	0	0
7	0	1	0	0	1	0	0	0	0	0	0	1
8	0	0	1	1	0	1	1	1	1	0	0	0
9	0	0	0	0	0	0	1	1	1	0	0	0
10	0	0	0	1	0	0	1	1	0	1	1	1
11	0	0	1	0	0	1	0	0	1	0	0	0
12	0	0	0	0	0	0	0	0	0	1	1	1

I created an Excel file of B so it is convenient to read the file instead of creating it in the RStudio. The column names are OD and the rows are edges we need.

In terms of X:
Actually, it is not hard for us to find that in the file “Road”, if those links are not related, they are all zero. So we can pick those useful 12 columns easily.

```
      A..B   B..A  B..C C..D   C..E   D..C  E..F    E..G   F..E   G..B   G..H H..G
[1,] 41313 101246 49721 7899 100975  59153 68601  136824 104450 109654  34441 7271
[2,] 40177 828327 45976 5400  86171  45595 51180  865384 830393 834126  36211 4953
[3,] 34597 517700 40700 5531  83767  48598 43384  560321 519938 523803  41709 5191
[4,] 34529 963985 43191 8207 226387 191403 42484 1150371 966468 972647 185294 7570
[5,] 29071 909595 35493 5782  71519  41808 33508  950499 912488 916017  39671 5189
[6,] 28950 795426 34439 5086  71106  41753 36622  832388 797904 800915  35885 4412
```

The dimension of the data is 96*12.

Fit the model

```
```{r}
tomo.fit <- tomogravity(data,B,0.01)

zhat <- tomo.fit$xhat#[,c(2,3,4,5,7,8,9,10,12,13,14,15)]
```

Here we set the tuning parameter to be 0.01.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12
1	2558.5253	31475.5130	6478.785	21763.196	31643.623	6514.475	75131.8694	5330.751	22484.328	4136.4169	293.86993	3612.2964
2	960.6506	18739.3741	4465.514	41780.611	7711.147	1837.652	777992.6700	11220.737	34213.678	6080.3090	87.69643	1710.5512
3	1017.2089	19193.2466	5797.751	39141.903	9879.711	2985.163	471813.0367	8204.584	35983.566	5492.8596	95.50104	1801.9269
4	572.8991	4009.9314	11008.326	158100.044	10625.915	29177.522	797809.1782	14127.910	147247.927	5365.6666	95.02072	664.9485
5	0.0000	0.0000	0.000	50468.677	0.000	2240.695	849918.1601	17325.191	37727.561	3909.5336	79.68430	0.0000
6	561.0155	6338.4855	3152.888	42547.535	3871.970	1926.201	744926.8731	12271.872	33726.805	4831.1785	79.55138	898.8514
7	3073.8357	14953.9968	11395.185	11080.796	15176.427	11564.598	12011.6935	3793.210	12536.679	2158.2660	681.56763	3315.5714
8	3089.9523	14412.7732	9659.207	11175.368	17389.698	11653.487	10402.3643	3252.838	10847.116	2226.3325	696.11380	3247.0917
9	2481.9950	25733.4668	12267.023	13138.340	16875.004	8043.930	21332.6257	2558.396	13061.672	1851.8632	222.02377	2302.0768
10	2609.9879	19687.0054	12159.987	14386.073	16810.516	10382.218	19752.8585	3052.284	14254.667	2165.0104	334.55692	2524.2732
11	3247.1741	25520.0001	12247.522	13147.787	17054.830	8185.744	20947.5787	3298.497	13041.721	2507.9996	394.77625	3102.7274
12	2497.3116	25129.9172	14257.045	14429.867	17694.579	10038.483	20698.5752	2570.954	14401.706	1866.7520	231.78486	2332.5372
13	3090.7577	6512.2383	17903.813	16862.312	20893.544	57442.605	4753.6156	3427.863	16193.503	2279.2958	1643.90949	3463.4178
14	1346.9114	2201.5767	25442.357	23462.933	25854.951	298812.940	1605.3512	2207.494	20445.492	1011.8473	1391.43417	2274.3007
15	619.6649	1167.6572	16852.350	167899.084	16377.961	236419.203	115084.4762	9421.922	162026.131	5115.2029	418.66741	789.2205
16	718.7951	1875.0409	14739.679	126398.320	16307.893	128202.668	120727.3866	8872.516	122426.789	5581.5935	410.22652	1070.1459
17	1524.4200	10665.1264	12636.129	48123.795	15313.430	18142.001	124339.1080	6565.164	46878.344	4992.1370	263.59734	1844.2252
18	2437.6815	14133.5935	11617.306	11588.803	15334.622	12604.699	10508.3849	2586.469	11429.087	1697.7488	417.78700	2422.8750
19	2290.7516	12998.3160	10612.412	9413.286	14141.639	11545.323	7512.7924	2183.012	9214.866	1362.4281	395.67232	2245.3682
20	2399.2029	14803.4404	11396.675	11368.036	15208.806	11710.183	10890.8451	2532.756	11218.125	1639.2630	381.29566	2352.6978
21	2923.0681	9715.9756	3014.561	8095.854	76280.217	23666.702	2601.7430	8171.501	7606.342	319.3711	1003.01322	3333.9533
22	4521.9648	3559.9157	7694.215	5695.969	11643.885	25161.626	1271.0798	3240.310	5616.511	2305.1603	5876.10630	4626.9181
23	1721.5105	2731.8785	6046.591	5520.183	10884.225	24088.823	1218.5594	1820.179	5317.265	785.7227	1173.65739	1862.6988

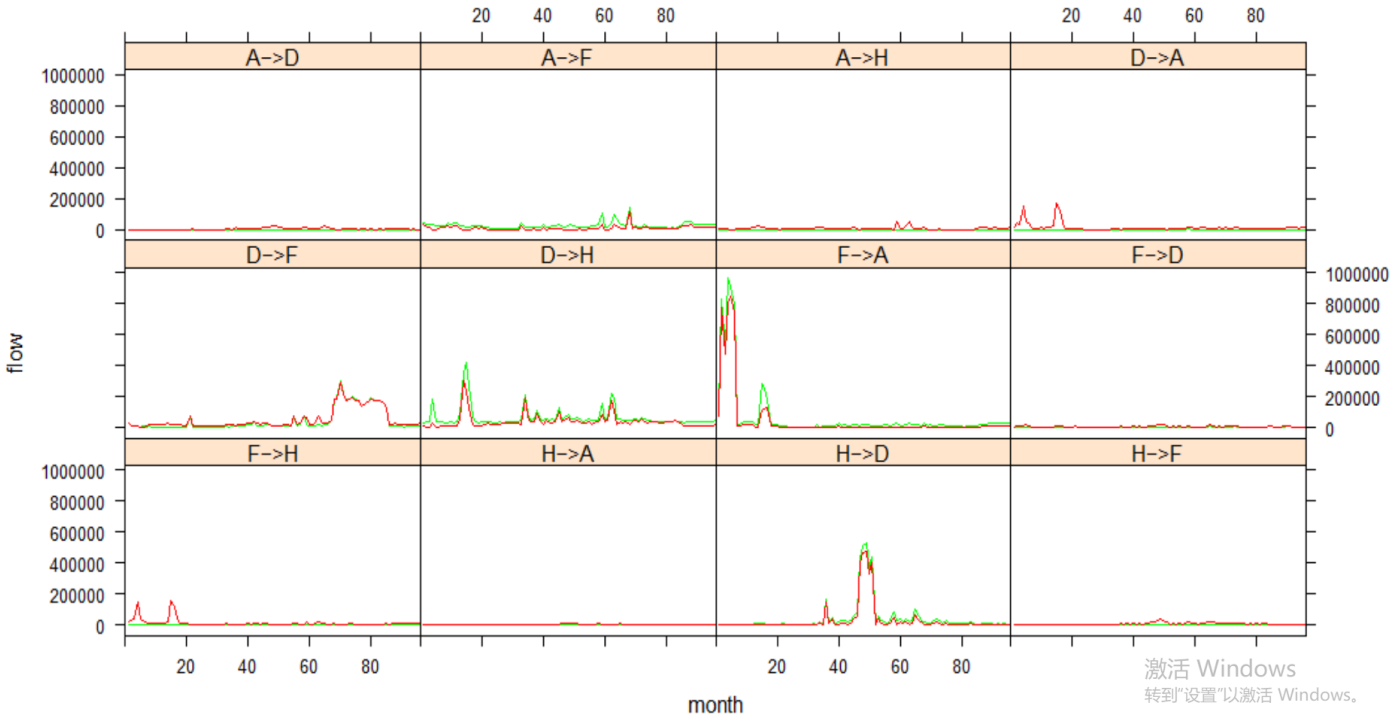
Here is the screenshot of the prediction. And in the next step, I will put the prediction and the actual
```

value together and compare them.

III Task II

Below is the comparation between the actual value and estimated value flow. The green one is the actual value and the estimated value is marked as red one.

From the pictures below, it is not hard for us to find that the estimation is very good if ignoring some outliers. Most waves share the similar shapes. So, it is a good model.



IV Appendix

---

title: "Network_Final"

author: "Chenrui Xu"

date: "2021/5/8"

output: html_document

```{r}

library(sand)

library(networkTomography)

library(igraph)

library(statnet)

```

```{r}

netmat <- rbind(c(1,2),

                c(2,1),

   c(2,3),

   c(2,9),

   c(3,4),

   c(3,5),

   c(3,11),

   c(4,3),
```

c(5,6),

c(5,7),

c(5,13),

c(6,5),

c(7,2),

c(7,8),

c(7,15),

c(8,7),

c(9,10),

c(10,3),

c(10,17),

c(11,12),

c(12,5),

c(12,17),

c(13,14),

c(14,7),

c(14,17),

c(15,16),

c(16,2),

c(16,17),

c(17,2),

c(17,3),

c(17,5),

c(17,7),

```
  c(17,9),
  c(17,11),
  c(17,13),
  c(17,15)
)
net <- network(netmat, matrix.type = "edgelist", directed = TRUE)
network.vertex.names(net) <-
c("A","B","C","D","E","F","G","H","I","J","K","L","M","N","O","P","Q")
gplot(net, vertex.cex = 3,
      label.pos = 5, arrowhead.cex = 1,
      displaylabels = TRUE, mode='fruchtermanreingold')
```

```{R}
path="C:/Users/StevenRui/Desktop/Network/Final"
Road=read.csv("C:/Users/StevenRui/Desktop/Network/Final/FinalProjectRoad.csv")
Transportation=read.csv("C:/Users/StevenRui/Desktop/Network/Final/FinalProjectTr
ansportation.csv")
```

```{r}
head(Road)
```

```{r}
data=data.frame(Road[,c(2,3,4,6,7,9,10,11,13,14,15,17)])
data=as.matrix(data)
head(data)
dim(data)
```

```{r}
dim(data)
```

```{r}
B=read.csv("C:/Users/StevenRui/Desktop/Network/Final/B.csv",head=F)
B=as.matrix(B)
dim(B)
# B2=rbind(
#     c(1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0),
#     c(0,0,0,0,1,1,1,1,0,0,0,0,0,0,0,0),
#     c(0,0,0,0,0,0,0,0,1,1,1,1,0,0,0,0),
#     c(0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1),
#     c(1,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0),
#     c(0,1,0,0,0,1,0,0,0,1,0,0,0,1,0,0),
#     c(0,0,1,0,0,0,1,0,0,0,1,0,0,0,1,0),
#     c(0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,1)
```

)

dim(B2)

```

```{r}

head(B)

```

```{r}

tomo.fit <- tomogravity(data,B,0.01)

zhat <- tomo.fit$Xhat#[,c(2,3,4,5,7,8,9,10,12,13,14,15)]

dim(zhat)

head(zhat)

View(zhat)

```

```{r}

Z=Transportation[,2:13]

nt=nrow(Z)

nf=ncol(Z)

t.dat=data.frame(z=as.vector(c(Z)),

zhat=as.vector(c(zhat)),t=c(rep(as.vector(Road$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)
z=as.vector(c(Z))
z=sapply(z,as.numeric)

xyplot(z~t|OD, data=t.dat,
panel=function(x,y,subscripts){
panel.xyplot(x,y,type='l',col.line='green')
panel.xyplot(t.dat$t[subscripts],
t.dat$zhat[subscripts],
type='l',col.line='red')
},as.table=T,subscripts=T,xlim=c(0,96),
```

xlab='month',ylab='flow')

```

```{r}

View(Transportation)

View(zhat)

```
```