CE19B058 Keshav Kr. Choudhary.

1. The 4 steps in the transport model are-

a) Trip generation: The city is divided into different zones and number of trips produced and attracted by ~~the~~ a zone is calculated based on different factors like people's income level, occupation, travel needs and habits, etc. The productions and attractions are listed in a PA matrix. Multiple linear regression model can be used for trip generation. The data required as input will be different independant factors with some numerical value associated to them which influence number of trips. Assumption for this model is that number of trips will be linearly dependant on the different factors that we choose. The choice of factors is also an assumption.

b) Trip distribution: The PA matrix obtained from trip generation is used, along with the cost ~~for~~ or impedance for different links, to get an origin-destination (OD) matrix. It gives us the number of trips between a given origin and destination. Gravity model is used for this purpose. The assumption is that, number of trips between zone i & j is directly proportional to P_i & A_j.

$$T_{ij} = \frac{P_i \left(A_j C_{ij}^{-d}\right)}{\sum_k \left(A_k C_{ik}^{-d}\right)}$$

c) Modal Split: Currently, we have the number of people who travel between any two zones. In this step, we find the mode of transport that they'll use for commute. Multinomial Logit Model is used for this purpose. Utility for different modes is calculated using utility function. A person chooses the mode with maximum utility. Utility depends upon trip, mode and individual characteristics. Some randomness is also associated. MNL calculates probability of choosing mode i among alternatives: $P(i) = \frac{e^{U_i}}{\sum_j e^{U_j}}$

The limitation of this model is the red bus - blue bus paradox, where number of people preferring bus over car will increase if half of the ~~buses~~ buses are painted red, and the other half blue, which is obviously wrong. A nested model can work better in such cases.

d) Trip Assignment: After getting the number of vehicles travelling from given origin to destination, we find the path that the vehicle will follow. Accordingly, we get a ~~demand~~ an ~~matrix~~ array which shows the number of vehicles on each link. One of the model used for this purpose is all or nothing assignment, where it is assumed that all the vehicles will choose the link which has minimum cost or impedance, and we assign all the traffic to that link. Limitation is that it doesn't account for the change in cost according to congestion. So, 2 a)

	HHVEHCNT	HHFAMINC	NUMADLT	DRVRCNT	HOMETYPE	HHSIZE	HH_0TO4	TELNUMCT	HHTRIPS
HHVEHCNT	1								
HHFAMINC	0.07600530564	1							
NUMADLT	0.3285512096	0.4964102985	1						
DRVRCNT	0.7128599441	0.02167108705	0.5843047258	1	1				
HOMETYPE	-0.2660662432	-0.1654954555	-0.1662094184	-0.4535394202	1				
HHSIZE	0.3693627814	0.09187701085	0.2714767782	0.3653383417	-0.3770801266	1			
HH_0TO4	0.1917529849	-0.1868117995	-0.06112274566	0.09656090992	-0.2022599587	0.5121933768	1		
TELNUMCT	0.2148678752	0.1894006287	0.2562500454	0.3180732126	0.2422718559	-0.07612991842	-0.1088931013	1	
HHTRIPS	0.3881717843	0.5491785454	0.4921935726	0.2279057911	-0.274571317	0.05398235417	0.1514584997	0.1209470955	1

SUMMARY OUTPUT

Other variables are ignored due to lower covariance

HHTRIPS = 0.8945352374 + 0.403501985*HHVEHCNT + 0.003532790098*HHFAMINC + 0.5905178399*NUMADLT -0.5507585354*DRVRCNT + -0.1442691947*HOMETYPE

Regression Statistics	
Multiple R	0.7239187545
R Square	0.5240583631
Adjusted R Squa	0.3988105639
Standard Error	0.4630596422
Observations	25

ANOVA

	df	SS	MS	F	Significance F
Regression	5	4.485939588	0.8971879176	4.184172229	0.009860325147
Residual	19	4.074060412	0.2144242322		
Total	24	8.56			

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%
Intercept	0.8945352374	0.4939243321	1.811077485	0.08596989849	-0.1392602546	1.928330729
X Variable 1	0.403501985	0.1760237795	2.292315197	0.03347532031	0.03507998599	0.7719239839
X Variable 2	0.003532790098	0.003286886374	1.074813576	0.295918584	-0.00334674204	0.01041232224
X Variable 3	0.5905178399	0.324459664	1.820003857	0.08455039648	-0.08858403092	1.269619711
X Variable 4	-0.5507585354	0.3267153671	-1.685744201	0.1081981642	-1.234581647	0.1330645762
X Variable 5	-0.1442691947	0.1033078418	-1.396498002	0.1786663448	-0.3604949893	0.07195659991

better alternative is incremental assignment.

C_{ii} is assumed to be 5, because $C_{ii} = 0$ gives ∞ paths.

2. b) For the given PA matrix, T_{ij} is calculated using gravity model and 10 iterations are performed by applying row-column factor method to balance out trips. C_{ij} matrix is used to get impedance values

c) After the addition of new links, the C_{ij} matrix will change. For the new impedances, again O-D matrix is calculated (10 iterations).

Shortest paths, using Dijkstra's algorithm.

	1	2	3	4	5	6	7	8	9
1		1→2 (5)	1→3 (4)	1→2→4 (10)	1→3→5 (9)	1→3→5 ↓ 6 (11)	1→3→5 ↓ 7 (13)	1→3→5 ↓ 8←7 (16)	1→3→5 ↓ 9←7 (17)
2			2→1→3 (9)	2→4 (5)	2→4→5 (7)	2→4→6 (8)	2→4→6 ↓ 7 (10)	2→4→6 ↓ 8←7 (13)	2→4→6 ↓ 9←7 (14)
3				3→5→4 (7)	3→5 (5)	3→5→6 (7)	3→5→7 (9)	3→5→7 ↓ 8 (12)	3→5→7 ↓ 9 (13)
4					4→5 (2)	4→6 (3)	4→6→7 (5)	4→6→7 ↓ 8 (8)	4→6→7 ↓ 9 (9)
5						5→6 (2)	5→7 (4)	5→7→8 (7)	5→7→9 (8)
6							6→7 (2)	6→7→8 (5)	6→7→9 (6)
7								7→8 (3)	7→9 (4)
8									8→9 (3)
9									

2 b) First Iteration for Tij:

Tij	1	2	3	4	5	6	7	8	9	Sum
1	4819.298963	6425.73195	4091.102758	2295.642994	2029.410931	1362.741885	646.1734136	169.5167107	160.3803947	22000
2	2150.341869	2867.122492	391.0364543	3822.829989	2521.457948	1407.731235	587.5632079	135.1215858	116.7952185	14000
3	3674.232643	1049.442392	1335.872844	604.3866185	586.1923602	421.7866899	210.995804	58.83221888	58.25842891	8000
4	79.18882615	394.0575326	23.21390123	525.4100434	3745.367173	903.1916573	255.4622337	42.99526393	31.1133685	6000
5	52.69425546	195.6411021	16.94755952	2819.214507	617.9482662	3171.616321	519.3305037	65.2137007	41.39378456	7500
6	55.42728378	171.0979606	19.10190142	1064.952786	4968.180418	784.0682206	338.8673015	57.03265352	41.27147512	7500
7	135.6064184	368.4689082	49.3036096	1554.169244	4197.413118	1748.440399	1510.829236	1450.091179	485.6778887	11500
8	110.9829059	264.3524882	42.88767865	816.0272075	1644.330619	918.0306084	4523.845095	623.2501724	2056.293225	11000
9	181.793383	395.6097178	73.52910824	1022.384533	1807.043684	1150.1826	2623.275874	3560.149726	1686.031374	12500
Sum	11259.56655	12131.52454	6042.995815	14525.01792	22117.34452	11867.78962	11216.34267	6162.20321	4677.215158	

10th iteration for Tij

```
1.0e+03 *

  3.8598    6.5671    3.3683    2.8279    2.0033    2.3584    0.6897    0.1162    0.1840
  1.5588    2.6521    0.2914    4.2623    2.2527    2.2050    0.5676    0.0839    0.1213
  3.1343    1.1423    1.1715    0.7930    0.6163    0.7775    0.2399    0.0430    0.0712
  0.0566    0.3592    0.0170    0.5773    3.2975    1.3942    0.2432    0.0263    0.0318
  0.0302    0.1432    0.0100    2.4876    0.4369    3.9317    0.3970    0.0320    0.0340
  0.0404    0.1589    0.0143    1.1925    4.4577    1.2334    0.3288    0.0355    0.0430
  0.0978    0.3393    0.0366    1.7248    3.7328    2.7261    1.4527    0.8957    0.5020
  0.0801    0.2433    0.0318    0.9053    1.4618    1.4309    4.3485    0.3849    2.1247
  0.1421    0.3946    0.0591    1.2292    1.7410    1.9428    2.7327    2.3825    1.8880
```

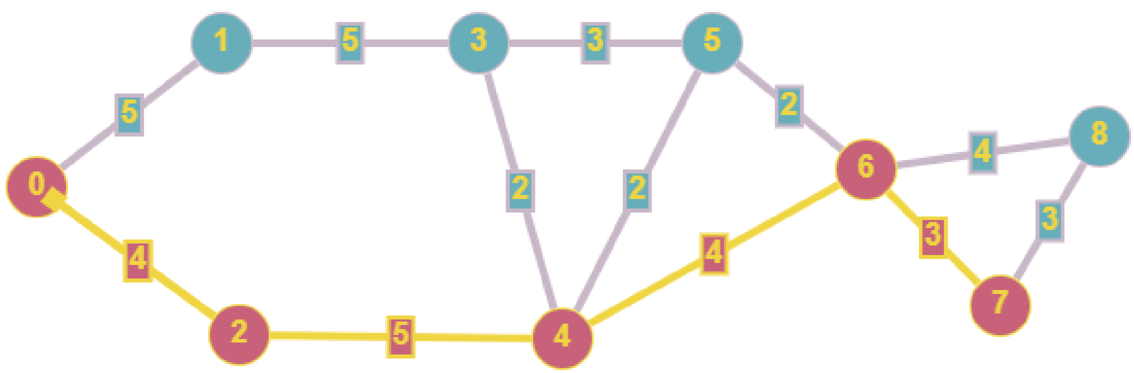
2 c)
1st iteration for Tij with new cost matrix

```
1.0e+03 *

  4.2842    5.7123    3.6369    2.0408    3.1163    1.9156    0.8523    0.2089    0.2327
  2.0702    2.7602    0.3765    3.6803    2.4274    1.6951    0.6779    0.1498    0.1626
  1.6722    0.4776    0.6080    1.0266    2.4319    1.1549    0.4378    0.0922    0.0990
  0.0693    0.3447    0.0758    0.4596    3.2760    1.3646    0.3160    0.0470    0.0470
  0.0887    0.1906    0.1505    2.7468    0.6021    3.0902    0.5060    0.0635    0.0616
  0.0594    0.1450    0.0779    1.2464    3.3662    0.5313    1.8514    0.1181    0.1044
  0.0912    0.2002    0.1019    0.9961    1.9025    6.3904    0.6848    0.6573    0.4756
  0.1274    0.2521    0.1223    0.8455    1.3621    2.3233    3.7475    0.5163    1.7034
  0.1777    0.3425    0.1643    1.0574    1.6532    2.5702    3.3936    2.1316    1.0095
```

10th iteration for Tij with new cost matrix

```
1.0e+03 *

  4.3976    6.4781    3.3628    2.2026    2.9100    1.5047    0.6949    0.1912    0.2740
  2.1174    3.1192    0.3469    3.9581    2.2588    1.3268    0.5508    0.1366    0.1908
  1.7922    0.5656    0.5870    1.1569    2.3712    0.9472    0.3727    0.0881    0.1217
  0.0773    0.4252    0.0762    0.5396    3.3278    1.1660    0.2802    0.0468    0.0602
  0.0983    0.2334    0.1503    3.2015    0.6071    2.6212    0.4455    0.0628    0.0784
  0.0658    0.1776    0.0777    1.4526    3.3944    0.4506    1.6300    0.1167    0.1327
  0.1075    0.2608    0.1082    1.2350    2.0409    5.7664    0.6414    0.6912    0.6433
  0.1428    0.3120    0.1234    0.9959    1.3882    1.9916    3.3345    0.5158    2.1888
  0.2010    0.4282    0.1675    1.2579    1.7017    2.2253    3.0499    2.1508    1.3102
```

Use of Dijasktra’s algorithm:



Matlab code used to calculate Tij in b and c part:

```
%% Calculating Ki

K = zeros(9, 1);
for i = 1:9
    for k = 1:9
        K(i) = K(i) + A(k)/(C(i, k)^1.9);
    end
end

%% Calculating Tij for first time using Gravity model

T = zeros(9, 2, 10);
for i = 1:9
    for j = 1:9
        T(i, j, 1) = P(i)*A(j)/(K(i)*C(i, j)^1.9);
    end
end

%% 10 iterations to compute Tij

rf = ones(9,1); cf = ones(9,1); %row and column factors
for k = 2:10
    if mod(k, 2) == 0    % Using row and column factor for
    for i = 1:9          % odd and even iterations respectively
        cf(i) = A(i)/sum(T(:, i, k-1));
        T(:, i, k) = cf(i)*T(:, i, k-1);
    end
    else
    for i = 1:9
        rf(i) = P(i)/sum(T(i, :, k-1));
        T(i, :, k) = rf(i)*T(i, :, k-1);
    end
    end
end
```

So, by all or nothing assignment, traffic on different links is:

1⟷2 8596 + 912 + 2280

2⟷4 4383 + 2492 + 1504 + 812 + 449 + 619 + 2280

1⟷3 5155 + 912 + 3008 + 1571 + 802 + 334 + 475

3⟷5 1233 + 2522 + 1025 + 481 + 212 + 289 + 3008 + 1571 + 802 + 334 + 475

4⟷5 1233 + 6529 + 2492

4⟷6 1504 + 812 + 449 + 619 + 2619 + 1515 + 1042 + 1318

5⟷6 1025 + 6016

5⟷7 802 + 334 + 475 + 481 + 212 + 289 + 2486 + 1451 + 1780

6⟷7 812 + 449 + 619 + 1515 + 1043 + 1318 + 7397 + 2108 + 2358

7⟷8 334 + 449 + 212 + 1043 + 1451 + 2108 + 4026

7⟷9 475 + 619 + 289 + 1318 + 1780 + 2358 + 3693

8⟷9 4340.

1-2	11788		4-6	9878
1-3	12257		5-6	7041
2-4	12539		5-7	8310
3-5	11952		6-7	17619
4-5	10254		7-8	9623
7-9	10532		8-9	4340