**Lesson 5**

**Dynamic** **Origin-Destination Demand Matrix Estimation (ODME) in DTALite**

*Please share your comments online for improving this document. If you have any questions or encounter any problems, please feel free to contact us (*[*jiangtao.liu@asu.edu*](mailto:jiangtao.liu@asu.edu)*;* [*xzhou74@asu.edu*](mailto:xzhou74@asu.edu)*). Your any feedback is greatly appreciated!*

**Data Set**: <https://github.com/xzhou99/learning-transportation/tree/master/Lessons/Lesson%205/Data%20Set>

**Learning Objective**:

1. Understanding dynamic OD demand matrix estimation implemented in DTALite;
2. Scenario evaluation in DTALite

This document aims to offer users to understand dynamic OD demand matrix estimation implemented in DTALite and further apply it for scenario evaluation. Section 1 illustrates the data requirement and setting process for performing ODME in DTALite. Section 2 shows the integration of ODME and specific scenario evaluation, which could be work zone, incident, ramp metering, etc., and further provide the guide about how to obtain the user equilibrium result under specific scenarios based on the estimated OD demand matrix/agent data. Section 3 introduces the general methodology of ODME implemented in DTALite.

The core estimation variable of ODME in DTALite is trip production of each zone. Based on the original travel demand in input\_demand.csv or the demand file users provide through input\_demand\_file\_list.csv, it is easy to calculate the trip production rate of each zone and the trip distribution ratio of each OD pair. During the ODME process, we fix the original trip distribution ratio of each OD pair and the time profile used to define the percentage of total travel demand departing at some specific time periods. After running the simulation, users can check the calibrated trip production of each zone through ODME\_ratio in file ODME\_final\_result.csv. In addition, the process of each iteration can also be checked in file ODME\_zone\_based\_log.csv and file ODME\_link\_based\_log.csv.

The key input and output files for ODME and scenario evaluation are listed in Table 1. The detail of some files will be explained in the following sections.

Table 1 Key input and output files for ODME

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **File Group** | **Input file list** | **Remark** | **Output file list** | **Remark** |
| **Demand** | 1. input\_demand.csv or input\_agent.csv | Demand content for specifying distribution | 1. output\_agent.csv | Final estimation results represented in terms of agent trajectory format |
| 2. input\_demand\_file\_list.csv | Demand file specification, specify temporal departure time distribution | 2. ODME\_zone\_based\_log.csv  3. ODME\_final\_result.csv | Zone based production based ratio; Iteration by iteration zone based total production adjustment |
| **Sensor** | 1. input\_link.csv | Store count\_sensor\_id, speed\_sensor\_id for referring sensor data | 1. ODME\_link\_based\_log.csv | Iteration by iteration link based flow adjustment |
| 2. sensor\_count.csv | Link based sensor count data |
| 3. sensor\_speed.csv (optional) |  | 2. debug\_validation\_results.csv | Link based simulated vs. observed results |
| **Scenario** | 1. input\_scenario\_settings.csv | ODME and assignment settings | 1. output\_summary.csv | Iteration-by-iteration UE and ODME statistics |
| 2. DTASettings.txt | Default settings for sequential ODME run |
| 3. scenario\_work\_zone.csv | Specify the capacity reduction scenarios in the estimation and prediction stages |

**1. ODME**

The general process of ODME is to adjust the given historical OD demand so that the final assignment results are consistent with observed link traffic measurements, such as, link count, link occupancy, link travel time etc. In addition to the basic traffic network data (input\_node.csv, input\_link.csv, input\_zone.csv, and input\_activitity\_location.csv), it also requires (1) a OD demand matrix seed as our initial demand values, (2) observed sensor data for calibration and (3) scenario settings files for algorithm performance. The demand seed could be a zone-to-zone demand file (such as, input\_demand.csv) or an activity-based demand file (such as, input\_agent.csv) with time profile in input\_demand\_file\_list.csv, and the sensor utilization involves sensor location information (input\_link.csv and sensor\_count.csv) and observed traffic data (sensor\_count.csv and/or sensor\_speed.csv). The specific settings need to be finished in input\_secenario\_settings.csv and DTASettings.txt, and if necessary, some scenario files also needs to be set up, such as, scenario\_work\_zone.csv. After ODME and traffic assignment, each vehicle’s travel information is stored in output\_agent.csv, which can also be treated as the estimated agent-based travel demand file. In addition, the link-based, zone-based and network-based statistics can be found in ODME\_zone\_based\_log.csv, ODME\_final\_result.csv, ODME\_link\_based\_log.csv, debug\_validation\_results.csv and output\_summary.csv.

**(1) Data preparation**

As described above, the required basic input data for ODME include the traffic network data (input\_node.csv, input\_link.csv, input\_zone.csv, and input\_activitity\_location.csv), original or historical travel demand data (input\_demand.csv and input\_demand\_file\_list.csv), and observed sensor data (sensor\_count.csv). The experiment is tested in the West Jordan traffic network, the data set of which can be download at [here](https://github.com/xzhou99/learning-transportation/blob/master/Lessons/Lesson%205/Data%20Set/1_ODME_West_Jordan.zip)

The detailed format of sensor\_count.csv is listed in Table 2, and sensor location information is also shown in input\_link.csv in Table 3.

Table 2. Sample data in sensor\_count.csv

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **count\_sensor\_id** | **from\_node\_id** | **to\_node\_id** | **day\_no** | **start\_time\_in\_min** | **end\_time\_in\_min** | **link\_count** | **speed** | **travel\_time\_in\_min** | **lane\_density** |
| 5010->4958 | 5010 | 4958 | 1 | 990 | 1050 | 49.5 |  |  |  |
| 4958->5010 | 4958 | 5010 | 1 | 990 | 1050 | 74.5 |  |  |  |
| 4952->5022 | 4952 | 5022 | 1 | 990 | 1050 | 221.5 |  |  |  |

Table 3. Sample data in input\_link.csv

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **link\_id** | **from\_node\_id** | **to\_node\_id** | **direction** | **length** | **number\_of\_lanes** | **speed\_limit** | **lane\_cap** | **count\_sensor\_id** | **speed\_sensor\_id** |
| 1285 | 1285 | 5018 | 1 | 0.2384 | 7 | 21 | 1428.6 | 1285->5018 |  |
| 1286 | 1286 | 11125 | 1 | 0.466 | 7 | 21 | 1428.6 | 1286->11125 |  |
| 1289 | 1289 | 4952 | 1 | 0.2427 | 7 | 21 | 1428.6 | 1289->4952 |  |

In this case, the observed link count from “link\_count” is used for ODME, and the start\_time\_in\_min and end\_time\_in\_min defines the corresponding observation time period. If optional density data or travel time data are also available, users can also prepare them in “lane\_density” or “ travel\_time\_in\_min” in this file. More importantly, to map sensors to links (in file input\_link.csv), one can use one of the following two methods to specify a link with sensors: (i) “from\_node\_id” and “to\_node\_id”, or (ii) field “count\_sensor\_id”, which should be first defined in file input\_link.csv. If not, warning messages will be issued.

The traffic network displayed in NeXTA is shown in Figure 1. Those green dots on some links represent the sensors’ location in the network.

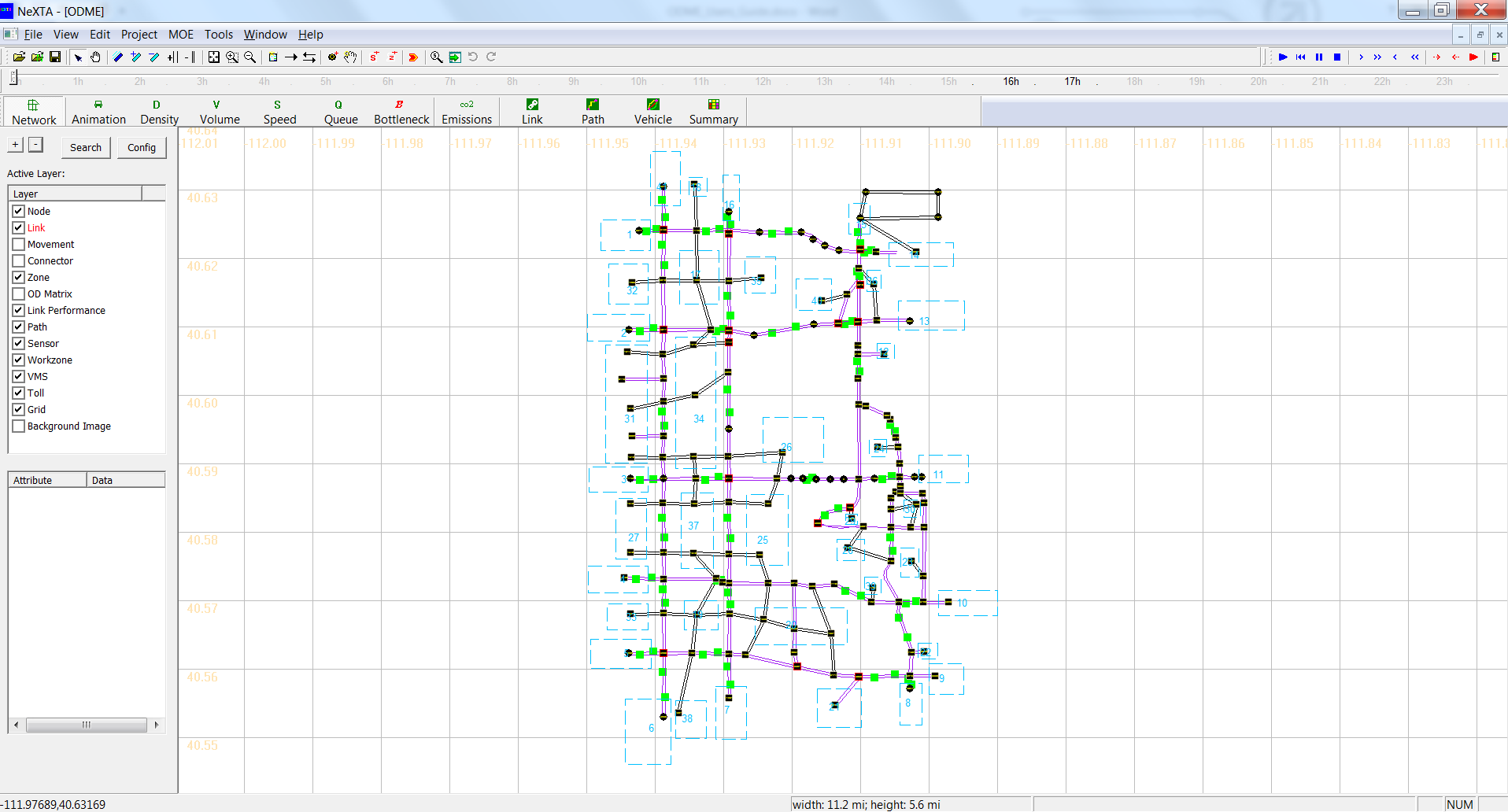


Figure 1. West Jordan traffic network in NeXTA

**(2) Parameters settings and ODME running**

The required parameter settings are defined in file input\_scenario\_settings.csv for varying traffic analysis purposes. Table 4 lists those important data fields for ODME and gives corresponding values and explanations for the West Jordan traffic model.

Table 4. Related attributes in file input\_scenario\_settings.csv for ODME

|  |  |  |
| --- | --- | --- |
| **Data Field** | **Value** | **Notes** |
| number\_of\_iterations | 50 | The total number of iterations for ODME |
| traffic\_flow\_model | 1 | This parameter defines a specific traffic flow model used in both assignment and ODME of DTALite; 1 indicates a point queue model in this example. The selection of Newell’s KW model is also feasible. |
| signal\_representation\_model | 0 | This parameter defines a specific signal control for DTALite. |
| traffic\_assignment\_method | 3 | This assignment method of “3” is dedicated to ODME |
| ODME\_start\_iteration | 20 | It defines the first iterative assignment period to converge to the user equilibrium state, and could generate a sufficient number of paths for path flow adjustment. The iteration number also indicate that ODME will begin at the 21th iteration. |
| ODME\_end\_iteration | 50 | It defines that ODME will end at the 50the itration. |
| ODME\_max\_percentage\_deviation\_wrt\_hist\_demand | 40 | The maximum percentage of demand deviation from base-line dynamic demand. |
| ODME\_step\_size | 0.05 | Moving size of each step in path flow adjustment algorithm |
| calibration\_data\_start\_time\_in\_min | 990 | This and the following parameter specify the time window for ODME to use the sensor data. Note that, users can prepare a long period of sensor data, say from 0 to 24 hours, but only use part of sensor data, say between min 990 and 1050, for calibration. |
| calibration\_data\_end\_time\_in\_min | 1050 |  |

**Remark**: in the DTASettings.txt file, you might see the following default settings:

[estimation]

number\_of\_iterations\_per\_sequential\_adjustment=10

time\_period\_in\_min\_per\_sequential\_adjustment=60

The time\_period\_in\_min\_per\_sequential\_adjustment defines the time period of each sequential adjustment in the algorithm as 60 min (1 hour). The number\_of\_iterations\_per\_sequential\_adjustment = 10 means that DTALite takes 10 iterations to adjust its demand during the time period of one sequential adjustment above (60 min or 1 hour). In this experiment, the calibrated demand period is 1 hour (990 min to 1050 min), so it requires 10 routing-simulation iterations to complete one global iteration for this demand period. To reach a reasonable converge, at least 3-10 global iterations are usually needed, so at least 30 iterations are required for the path flow adjustment for ODME. Meanwhile, with the starting 20 iterations for reaching the user equilibrium state, a total of 50 iterations will be run for this experiment.

In addition, in order to improve the computational efficiency, users can change the value of the max\_number\_of\_threads\_to\_be\_used in DTASettings.txt as what you want based on your computer configuration to perform parallel computing in DTALite. After ensuring that all data preparation and parameter settings are ready, you can run DTALite directly or call it through the interface of NeXTA.

**(3) Result analysis**

Though checking output\_summary.csv file, users can better understand the process of ODME in DTALite. For the first 20 iterations, a standard dynamic user equilibrium method, MSA, is used. It is expected to see the UE gap (Avg User Equilibirum (UE) gap (min) and Relative UE gap (%)) dramatically decreases and finally reach a stable state, which is shown in Figure 2. In addition, Figure 3 gives the general statistics about different traffic measures for the first 20 iterations

Figure 2. The trend of average UE gap and relative UE gap

  
Figure 3. The statistics of different traffic measurements for the first 20 iterations

In the following 30 iterations, users can check the R\_squared values from the iterative adjustment process, which should show an increasing pattern toward a reasonable statistics of 0.7, 0.8, or 0.9 in output\_summary.csv. The other measures related to ODME include the link count estimation absolute and percentage errors. The summary result of the following 30 iterations is shown in Figure 4, where the R\_squared values are marked in the red rectangle.



Figure 4. The summary result of the following 30 iterations

The comparison between simulation link measurements and observed link measurements is available in file debug\_validation\_results.csv. Figure 5 (a) illustrates the comparison result in link counts. Users can also check the comparison result in NeXTA through clicking “Tools”🡪 “Sensor Data Management” 🡪 “View Validation Plot for Link Count”, shown in Figure 5(b).



Figure 5. Comparison result between observed link count and simulated link count after ODME

**2. Scenario analysis using estimated OD demand in one simulation**

For performing scenario analysis, such as, work zone, incident, ramp metering, etc., the travel demand should be calibrated in advance. In this section, the integration of the two parts above is realized in DTALite through just one simulation, the process of which can be illustrated in Figure 6.



Figure 6. The process of one simulation for scenario analysis using estimated OD demand

At stage 1, the user equilibrium is reached on the basis of traffic network data and historical OD demand and the route choice set is generated for the path flow adjustment at stage 2, where observed sensor data is input and used for OD demand estimation. Based on the estimated OD demand, the scenario analysis is performed for traffic state prediction or new user equilibrium condition searching.

**(1) Evaluate specific scenarios using estimated OD demand**

The experiment is still the West Jordan traffic network and its data set can be download at [here](https://github.com/xzhou99/learning-transportation/blob/master/Lessons/Lesson%205/Data%20Set/2_ODME_Scenario_1_West_Jordan.zip). In this case, it is assumed that the work zone occurs on links 5112🡪5589 and 5589🡪5114. The detailed data are input in file Scenario\_Work\_Zone.csv shown in Table 5.

Table 5. Input data of Scenario\_Work\_Zone.csv

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Link** | **Scenario No** | **Start Day** | **End Day** | **Start Time in Min** | **End Time in min** | **Capacity Reduction Percentage (%)** | **Speed Limit** |
| [5112,5589] | 0 | 51 | 51 | 0 | 1440 | 50 | 50 |
| [5589,5114] | 0 | 51 | 51 | 0 | 1440 | 50 | 50 |

As listed in Table 5, both “Start Day” and “End Day” have same day value of 51, which indicates that the work zone will happen at the 51th iteration and the first 50 iterations is the process of ODME in Section 1. As a result, the vaule of “number\_of\_assignment\_days” should be set as 51 in file input\_scenario\_settings.csv.

The simulation result of the 51th iteration in output\_summary.csv is shown in Figure 7. It is observed that the average speed decreases and the average travel time increases due to the link capacity and speed limit reduction in Scenario\_Work\_Zone.csv.



Figure 7. Summary of the 51th iteration for work zone scenario

**(2) Find the user equilibrium state for specific scenarios**

DTALite can also continue to run to obtain the user equilibrium condition for specific scenarios through just one simulation. We still choose the West Jordan traffic network as our test case, which can be downloaded at [here](https://github.com/xzhou99/learning-transportation/blob/master/Lessons/Lesson%205/Data%20Set/3_ODME_Scenario_2_West_Jordan.zip). The updated file Scenario\_Work\_Zone.csv is listed in Table 6.

Table 6 Updated input in Scenario\_Work\_Zone.csv

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Link** | **Scenario No** | **Start Day** | **End Day** | **Start Time in Min** | **End Time in min** | **Capacity Reduction Percentage (%)** | **Speed Limit** |
| [5112,5589] | 0 | 51 | 70 | 0 | 1440 | 50 | 50 |
| [5589,5114] | 0 | 51 | 70 | 0 | 1440 | 50 | 50 |

As shown in Table 6, “Start Day” and “End Day” are set as 51 and 70, respectively, which means that DTALite will run 20 more iterations for reaching the user equilibrium condition under work zone scenario after the first 50 iterations for ODME. Therefore, the vaule of “number\_of\_assignment\_days” will be set as 70 in file input\_scenario\_settings.csv. The general summary result of the last 20 iterations is shown in Figure 8, and one stable user equilibrium condition is observed in this case.

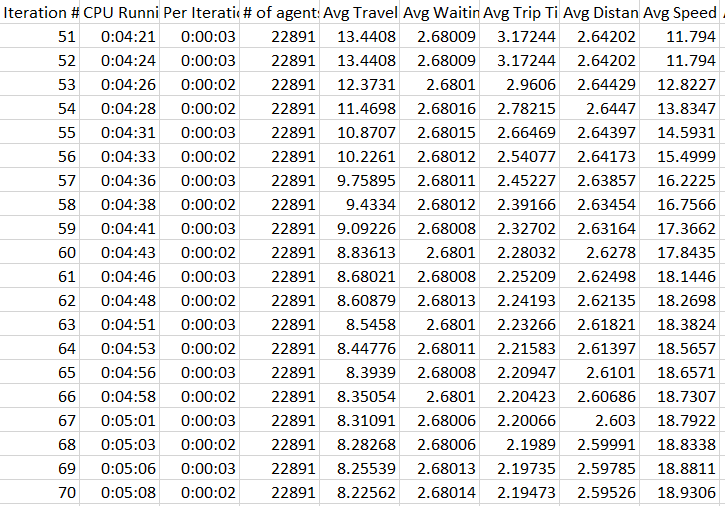


Figure 8. Summary of the last 20 iterations for work zone scenario

**3. Methodology**

**3.1 Introduction**

ODME implemented in DTALite is based on a single-level nonlinear optimization model proposed by Lu, Zhou and Zhang (2013). The general introduction on OD demand estimation problem can be found at [here](https://docs.google.com/document/d/1a10qMChKAz5u1YLg1nx9g8eVJv2HH24Z9ux0XJHgSpI/edit?pref=2&pli=1).

Our model has the following key features:

* The model is a path flow-based optimization model, which incorporates heterogeneous sources of traffic measurements and does not require explicit dynamic link-path incidences.
* The objective is to minimize (i) the deviation between observed and estimated traffic states and (ii) the deviation between aggregated path flows and target OD flows, subject to the dynamic user equilibrium (DUE) constraint represented by a gap-function-based reformulation.
* A Lagrangian relaxation-based algorithm which dualizes the difficult DUE constraint to the objective function is proposed to solve the model.
* This algorithm integrates a gradient-projection-based path flow adjustment method within a column generation-based framework.
* DTALite, a dynamic network loading (DNL) model which is based on Newell’s simplified kinematic wave theory, is employed in the DUE assignment process to realistically capture congestion phenomena and shock wave propagation.
* This optimization also derives analytical gradient formulas for the changes in link flow and density due to the unit change of time-dependent path inflow in a general network under congestion conditions.

Reference: Lu C-C, Zhou\*, X. Zhang, K. (2013) Dynamic Origin-Destination Demand Flow Estimation under Congested Traffic Conditions. Transportation Research Part C. 34, 16-37. ([A short version of the paper](http://www.civil.utah.edu/~zhou/Single-level_path-flow-OD-estimation_Zhou-Lu-Zhang.pdf))

**2.2 Mathematical model and solution algorithm**

Given sensor data (i.e. observed link flows and densities) and target (aggregated historical) OD demands, the proposed single-level time-dependent path flow estimation model is a nonlinear program with the path flows and least path travel times as the decision variables. Denote , and . The objective function, Eq.(1), minimizes the weighted sum of the deviation between estimated time-dependent OD demands (or aggregated path flows) and target demands and the deviation between estimated and observed link flows and densities, where , and are the weights reflecting different degrees of confidence on target OD demands and observed link flows and densities, respectively.

**P1: Nonlinear program**

(1)

Subject to

(2)

(3)

(4)

(5)

(6)

where

: set of links

: set of OD pairs

: set of paths

: set of links with sensors,

: set of discretized departure time intervals

: set of discretized observation time intervals

**Index:**

: index of simulation time intervals, .  This paper refers to any particular time interval as the time

: index of departure time intervals,

: index of OD pairs,

: index of paths for each OD pair,

: index of links,

**Traffic measurements inputs**

: observed number of vehicles passing through an upstream detector on link during observation interval

: observed density on link during observation interval

: target demand, which is the total traffic demand for OD pair over a planning horizon

**Estimation variables**

: estimated path flow on path of OD pair and departure time interval

: estimated path travel time on path of OD pair and departure time interval

: estimated least path travel time of OD pair and departure time interval

: estimated number of vehicles passing through an upstream detector on link during observation interval

: estimated density on link during observation interval

: estimated demand of OD pair and departure time interval

**Solution algorithm**

This section describes the Lagrangian relaxation-based heuristic for solving the single-level time-dependent path flow estimation model. We propose the following heuristic solution method to efficiently obtain good solutions for problem instances on road networks of practical sizes. The heuristic integrates Lagrangian relaxation and column generation methods to solve the time-dependent path flow estimation model, P1. The gap function constraint Eq.(3) is relaxed to the objective function Eq.(1) with a non-negative Lagrange multiplier l. The resulting Lagrangian subproblem is given as follows.

: (7)

Subject to constraints (2), (4), (5) and (6),

where and are the vectors of path flows and least path times respectively. For a given l, the solution to provides a lower bound to . The Lagrangian dual problem is given as follows.

:

Subject to (8)

The heuristic consists of two major algorithmic steps: at each iteration n, (i) given a Lagrange multiplier , find an optimal path assignment and least path travel times by solving the Lagrangian subproblem, **P2**, and (ii) given a vehicle path assignment and least path travel times, update the Lagrange multiplier by using the following rule.

(9)

where is the step size for updating the Lagrange multiplier.

Accordingly, this heuristic has two loops.  The outer loop is for updating the Lagrange multiplier using the rule described in Eq.(9). For each outer loop iteration n (i.e., corresponding to a given Lagrange multiplier ), a column generation-based approach is used to solve the Lagrangian subproblem . This approach forms an inner loop for solving a DUE assignment problem under a restricted feasible solution space. In each inner loop iteration m, a time-dependent shortest path algorithm (Ziliaskopoulos and Mahmassani, 1993) is adopted to generate time-dependent least time paths and to augment the restricted path set. In light of the time-dependent shortest path algorithm, the least path travel times are obtained to satisfy the constraints Eq. (4) and (5), thus, these definitional constraints of the least travel times can be dropped in solving the restricted Lagrangian subproblem. To solve the restricted subproblem, a gradient-projection-based descent direction method (Lu et al., 2009) is used to update path flows , while maintaining the feasibility of  non-negativity constraints Eq.(6). Specifically,

(10)

where is the step size, and the gradients, which consist of the first-order partial derivatives with respect to a path flow variable , can be derived as follows.

(11)

(12)

(13)

(14)

Estimated link flows, densities, and link/path travel times and the corresponding partial derivatives, namely, and are obtained from the DNL model presented in Section 2.

The steps of this algorithm are presented as follows in Figure 9.

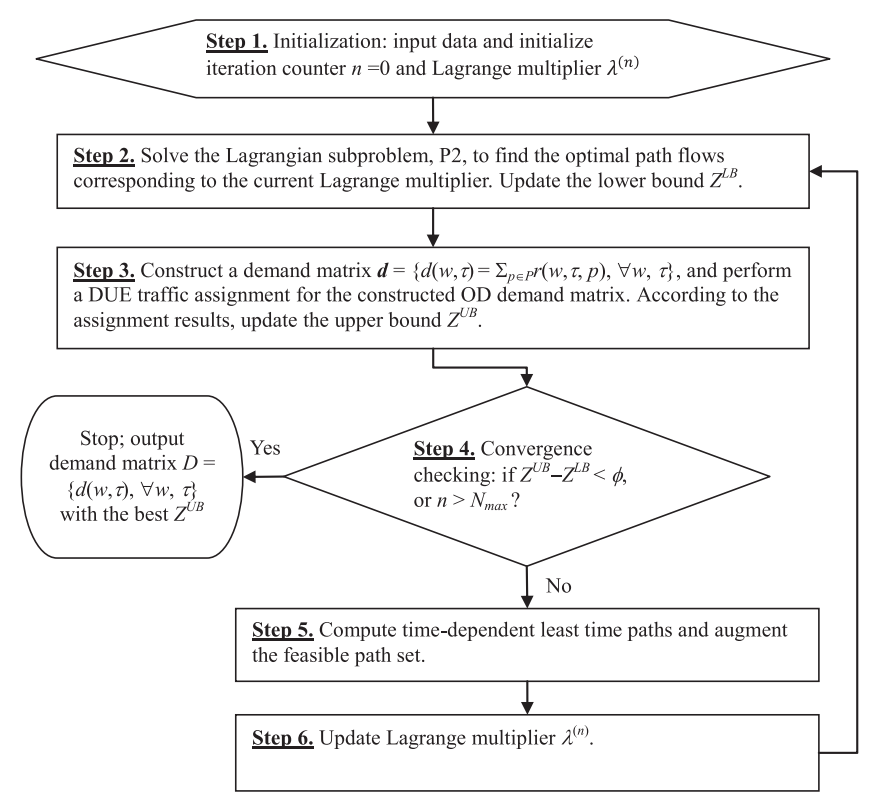


Figure 9. The procedure of proposed algorithm

Typically, a Lagrangian solution framework requires obtaining exact solutions to relaxed subproblems. It should be remarked that, analyzing existence and uniqueness of solutions to the DUE problem for multiple OD pairs are very challenging, and the gradient-based algorithm through Eqs. (10)-(14) cannot guarantee that the relaxed (nonlinear) problem P2 is solved to its optimality. Thus, when no global optimum solution is available for P2, the proposed overall Lagrangian solution algorithm is still a heuristic method in nature.

Solving the proposed single-level dynamic OD estimation model requires the evaluation of the partial derivatives with respect to time-varying path flows, i.e.,, and . These partial derivatives represent the marginal effects of an additional unit of path inflow on link flow and density and path travel time. This section delineates the evaluation of these partial derivatives due to path flow perturbation in a congested network, based on cumulative link inflow and outflow curves. The following notation is used throughout this section.

: the number of links on the path

: link index

: the time when an additional unit of perturbation flow arrives at link

: the time when an additional unit of perturbation flow departs at link

: the time when the queue starts to form on link

: the time when the queue vanishes on link

:

: the time when the queue on link starts to spillback to its upstream link

cumulative arrivals at time

: cumulative arrivals at time

The following propositions can be directly induced from Figure 10 for deriving the marginal effects on link flow (inflow and outflow) and density.

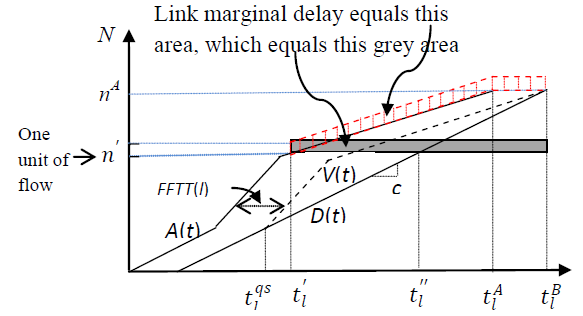


Figure 10. Impact of adding a unit of flow on a congested link

**Proposition 1**: Under *free-flow* conditions, an extra unit of flow arriving at the upstream end of link *l* at time : results in the following: (i) the link inflow and outflow increase by at times and , respectively, and the flow rates at other time intervals do not change; (ii) the link density increases by from to ; (iii) the individual travel times are not changed, and

**Proposition 2**: Under *partially* congested conditions and constant link (outflow) capacity , an extra unit of flow arriving at the upstream end of link at time results in the following: (i) the link inflow and outflow increase by at times and , respectively, and the flow rates at other time intervals do not change; (ii) the link density increases by 1 from and ; (iii) the flows arriving between and experience the additional delay , because it takes to discharge this perturbation flow.