CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Dynamic Origin/Destination Estimation Using True Section Densities

Carlos Sun
California PATH

California PATH Research Report
UCB-ITS-PRR-2000-5

Report for MOU 377

March 2000

ISSN 1055-1425

CALIFORNIA PARTNERS FOR ADVANCED TRANSIT AND HIGHWAYS

Dynamic Origin/Destination Estimation Using True Section Densities

FINAL REPORT

Carlos Sun

California PATH

Himanshu Porwal

University of California, Irvine

Prepared for

Partners for Advanced Transit and Highways

PATH MOU 377

August 1999

ACKNOWLEDGEMENT

The authors would like to thank Professor Jayakrishnan for the immeasurable amount of help and inspiration he provided on this project. The authors would also like to thank Professor Will Recker and Professor Amelia Regan for their useful suggestions during the entire course of this research. In addition, the authors would like to thank Jun Oh, Dr. Anthony Chen, Craig Rindt, and Anup Kulkarni for providing valuable insights. And last but not least, the authors would like to thank Ziggy Bates for assisting with computer equipment.

ABSTRACT

This final report presents a practical approach for dynamic origin/destination demand estimation. The proposed dynamic origin/destination estimation framework addresses many of the shortcomings of the existing formulations and presents a formulation for general networks and not just corridors. One unique feature of this framework is its use of section density as a variable instead of flow. The framework is built upon the foundation of static origin/destination matrix estimation by adding the temporal aspect. Two traffic assignment models, namely DYNASMART and DTA are used for assigning dynamic ODs onto the network and 1-Step Kalman Filter and Least Squares methods are used for optimizing the errors between the estimated and the true section counts. 1-Step Kalman Filter is considered as a special case of a Kalman Filter which is developed for future work with a rolling horizon estimation framework. In addition, this formulation also describes an infrastructure from which real-time traffic counts and other section data on various freeways could be collected and used in dynamic frameworks.

Keywords: Dynamic O-D Estimation/Prediction, Advanced Traffic Management System, Advanced Traveler Information System, Traffic Assignment

EXECUTIVE SUMMARY

One of the objectives of this research is to formulate a dynamic origin/destination estimation framework that is both theoretically sound and usable in the California PATH/University of California, Irvine (UCI) Testbed program. Another objective is to develop a framework that can operate off-line for evaluation of ATMIS strategies and in the future, adapt it for on-line usage. A third objective is to build a model upon the foundation of static origin/destination demand matrix estimation by adding temporal aspect. Lastly, this research seeks to perform sensitivity analysis to study the effects of changes in variables on the overall performance of the model.

The dynamic origin/destination framework is composed of two interacting modules: DOE (Dynamic Origin/Destination Estimation) and DTA (Dynamic Traffic Assignment) or a simulation model (DYNASMART). The DOE will estimate the OD matrix that generated observed section densities. The DTA in turn uses these demands and returns to the ODE newly computed section densities and a dynamic assignment matrix. The equilibration between the two modules will minimize the discrepancies between observed section densities and the ones computed by the framework.

The ability of the Testbed to generate real-time traffic data has been exploited in this research. Both measured and simulated data have been utilized to develop and test the resulting framework. Initial investigation focuses on small experimental networks.

These include grid networks that represent the characteristics of actual urban networks. Larger networks have been used to study computational issues.

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

CHAPTER 1. OVERVIEW

1.1 BACKGROUND AND PROBLEM STATEMENT

Growing concern about urban congestion prompted the U.S. department of Transportation to create the Intelligent Vehicle Highway Systems (IVHS) program, now officially called the Intelligent Transportation Systems (ITS) program, as part of the Intermodal Surface Transportation Efficiency Act of 1991 (ISTEA). ISTEA's stated mission was "to develop a National Intermodal Transportation System that is economically efficient and environmentally sound, provides a foundation for the nation to compete in the global economy, and will move and goods in an energy efficient manner" (U.S. Department of Transportation, 1991).

With the recent technological advances in communication, automation, electronics, computing and information, ITS seeks to apply and integrate these emerging technologies to combat traffic congestion, enhance mobility, eliminate wasteful travel, reduce energy consumption and environmental impact, and improve traffic safety. Within the general framework of ITS, Advanced Traffic Management Systems (ATMS) and Advanced Traveler Information Systems (ATIS) are two systems that can improve traffic conditions by managing the transportation systems more efficiently and giving system users better information. ATMS employ these innovative technologies and integrate various management and control strategies to bring order and efficiency to the movement of highway vehicles while ATIS attempt to make travel easier and safer by providing users with information on traffic conditions, routes, and schedules.

At the heart of Advanced Transportation Management and Information Systems (ATMIS) is the dynamic origin/destination demand model. The problem of dynamic origin/destination demand estimation is vital to all aspects of ATMIS research. This estimation problem involves the determination of the number of trips going from an origin to a destination during a given time step given the measured traffic on the network links. Any traffic assignment algorithm is dependent on accurate origin/destination demands as input. ATMIS rely on dynamic origin/destination demand to provide reliable short-term predictions regarding the state of the traffic systems in order to implement different management and information strategies to improve network performance. Applications that can use these short-term traffic forecasts include route guidance systems proposed for ATIS, traffic signal and freeway ramp controls as part of the control strategies in ATMS, and other schemes such as dynamic congestion pricing and Automated Highway Systems (AHS). Route guidance systems influence travel choices to better utilize the existing network capacity. Traffic signal and freeway ramp controls achieve system-wide reduction in total travel delay.

The dynamic origin/destination estimation problem studied in this research is intended to support applications in both ATMS and ATIS. Given the time-invariant traffic network characteristics (i.e., geometry, link connectivity, link length, free-flow speed, and capacity) and time-dependent true section measures (link densities), the proposed research finds the dynamic origin/destination demand matrix that resulted in the observed true section measures (link densities) for the network. Kalman Filter and Least Squares methods have been used in this research to track the demands and to minimize the error between estimated and observed section densities.

1.3 FINAL REPORT OUTLINE

This final report is organized into eight chapters. After the overview given in chapter 1, chapter 2 provides a brief history of dynamic origin/destination demand estimation research by highlighting some of the major developments in that field. In addition, this chapter provides an extensive dynamic OD estimation review, classifying the different models into two approaches.

Chapter 3 discusses the ability of UC Irvine Testbed for generating real-time traffic data for research purposes, as well as the basic concepts of the distributed computing platform, CORBA (Common Object Request Broker Architecture). It also explains some of the concepts that are crucial to the development of any Client/Server based application. A short discussion on the real data is also provided.

Chapter 4 discusses some basic concepts in DTA. It describes the approach developed for dynamic OD estimation using DTA. It presents concepts in Kalman Filter and Least Squares methods for optimization. It also formulates the problem solution using Kalman

Filter and the Least Squares method. The results for both the optimization methods are reported and compared.

Chapter 5 discusses concepts in DYNASMART and the dynamic OD estimation problem formulation using Kalman Filter and Least Squares methods. The results are reported and compared to the results obtained using DTA for traffic assignment.

Chapter 6 gives the details of the sensitivity analysis of the model developed. Changes in the performance of the model due to changes in various variables are presented. Also, the effect of changing a variable on the model’s performance with time is studied.

Chapter 7 identifies different avenues for future research. It discusses various non-traditional approaches for dynamic OD estimation problem formulation. It also introduces the concepts of the Rolling Horizon framework formulation.

Chapter 8 summarizes the findings of this final report and presents the conclusions of this research.

CHAPTER 2. LITERATURE REVIEW

2.1 BACKGROUND AND LITERATURE REVIEW

The problem of origin/destination demand estimation is vital to all aspects of ATMIS (Advanced Transportation Management and Information Systems) research. An origin/destination demand matrix is a vector with components that denote the average number of trips going from an origin to a destination. Traditionally, the method for obtaining an origin/destination matrix for large scale networks is to employ the use of household surveys coupled with roadside surveys. However, this method is expensive and also not feasible for real-time applications. Therefore, there has been research in the static origin/destination demand matrix estimation using network-wide link traffic counts and combining these with other available information. One source of information that we have already mentioned is household activity surveys, another source is the knowledge of the a priori probability distribution of the origin/destination matrix. Yet another is the use of previously estimated origin/destination matrices.

When a research area is considered, it is divided into several zones or centroids. The static origin/destination demand matrix consists of all the trips from all the origin zones to all the destination zones. In general, the number of origin/destination pairs are greater than the total number of links. This means that by just using traffic counts, the estimation problem is under-specified and there is no unique solution. For this reason additional information is needed to determine a unique origin/destination demand matrix.

There are several methods for formulating estimators for an unique matrix. In transportation and regional planning, the most popular approach is the maximum entropy model (Van Zuylen and Willumsen, 1980). The assumption of this approach is that all of the combinations of individual travel decisions, so called states, are equally likely to occur. The set of origin/destination flows with the highest likelihood of occurring is therefore the set with the maximum number of states. The conventional method uses this maximum entropy consideration to obtain a doubly-constrained gravity model.

Another method is classical statistical inference techniques. The two main estimators are the maximum likelihood and the generalized least-squares. The maximum likelihood estimator maximizes the likelihood of observing the experimental data condition on the true trip matrix. For this method, distributional assumptions need to be made for the sample and traffic counts. On the other hand, no distributional assumptions need to be made for the generalized least squares approach.

The last method is the Bayesian one, which uses a priori probabilities on the trip demands. By combining these probabilities with the conditional probability on the traffic counts, one can obtain the posterior probability of the demand conditioned on the traffic count. The arguments of this probability can then be maximized by different methods.

Recently, there has been increased research in the area of dynamic origin/destination demand matrix estimation. There has been a number of procedures proposed. Most of them assume that the time taken by vehicles to traverse the network under consideration is smaller than the time step of the procedure. In the case of an intersection, origin/destination demand matrix estimation becomes the problem of finding the turning proportions. A simple way of classifying dynamic origin/destination demand estimation is to separate the approaches into two camps: assignment-based and non-assignment based.

The non-assignment based approach by definition does not use dynamic traffic assignment. They estimate origin/destination parameters directly from time-series input/output flows. Sometimes, prior origin/destination demand matrices are used for initialization. Cremer and Keller (1987) highlighted the causal relationships that exist between the time variable sequences of entrance flow volumes and the sequences of short-time exit flow counts. They claim that enough information can be obtained from the counts at the entrances and the exits to obtain unique and bias-free estimates for the unknown O-D flows without further a priori information. They proposed four methods: which were the ordinary least squares estimator, constrained optimization, Kalman filter, and “simple recursive” formulation. Nihan and Davis (1987) proposed using recursive prediction error techniques and input/output volume counts. They discussed different search methods and applied them for a corridor where route choice between origin and destination can be ignored. The complicated problem of route choice behavior is avoided and a relatively simple linear model has been used to relate the input counts to the output counts. They concluded that Gauss-Newton appear superior to gradient-based search methods while the constraining procedures improve accuracy. Van der Zijpp and Hamerslag (1994) augmented previous formulations by incorporating inner-link induction loop data. Based on assumed constant split ratios and a trip generation model, they proposed a Kalman-based method for freeway corridors that uses the model-predicted link-flow variances and covariances while processing the measurements. They also proposed a simple but effective solution to the problem of initializing the Kalman filter and imposing natural constraints to the estimates. The resulting method was tested on both the simulated and observed data and compared it with other methods such as least squares and constrained optimization, and concluded that Kalman based method leads to superior results.

One critical reference, Van der Zijpp (1997), added partial origin/destination information by using Automatic Vehicle Identification (AVI) technology. An example of an AVI technology is license plate recognition based on image processing. They proposed a method to track time-varying traffic patterns from a combination of link volume counts and trajectory observations obtained from induction loops and AVI equipment at arbitrary (but fixed) locations. They applied the approach to a single motor-way corridor with no route choice alternative and proposed the Bayesian updating scheme that used multivariate normal and truncated multivariate normal assumptions for the subjective probability distributions. The advantage of this new procedure is that it deals with the inequality constraints in an appropriate statistical manner. Bell (1991) reviews a number of procedures for the estimation in real time of origin/destination flows in small networks which are mostly based on the assumption that the time taken to traverse junction or network is either small in relation to the chosen time interval or is equal to the fixed number of time intervals. In reality, there will be a distribution of travel times that may span a number of time intervals. He proposes two ways to allow for this in the estimation of origin/destination flows. The first, appropriate if travel times are approximately geometrically distributed, makes use of the recurrent model of platoon dispersion. The second makes no assumption about the form of the distribution of travel times, but in general requires the estimation of substantially more parameters. Bell et al. (1995) partitioned travel time for each link into delay and undelayed travel time. The links were assumed to be of two types. For the first type of link, an external estimate of flow and travel time over the estimation interval is provided. The second type of link is characterized by a finite capacity, and delay is incurred where the demand would otherwise be in excess of the capacity. Demand is determined by a logit route choice model and an equivalent convex programming problem is formulated and an iterative solution procedure is set out. Traffic is assigned to paths according to a logit model on the basis of the path travel times and the dispersion parameter so as to reproduce the traffic flow measurements. Diagnostic procedures and the incorporation of prior information on the relative magnitudes of origin/destination movements were considered. He proposed a non-linear programming Path Flow Estimator (PFE) to estimate stochastic user equilibrium (SUE) path flows.

Chang and Wu (1995) presents an innovative method for estimating the dynamic network O-D matrices with time series of links and screen-line flows. They proposed a model that takes full advantage of the available link flow information, and increases the observability of the dynamic inter-relations, through arbitrary selected screen-lines which reside in the path of an origin/destination pair, between network O-D patterns and the resulting link flow distributions. The resulting screen-line flows are estimated from normal distribution assumptions and a mean arrival time.

The assignment-based approach depends on a reliable and descriptive dynamic model for generating the network link flow usage pattern. Camus et al. (1994) added historical information and combined it with traffic counts at on-ramps. They presented an approach for real-time prediction of Origin Demand matrices per time slice and applied their approach to an Italian motor-way. They concluded that the prediction procedures based on both the direct approach (where O-D matrices per time slice are directly observable) and indirect approach can perform equally well. Ashok and Ben-Akiva (1993) builds upon Okutani's work (1987) by formulating the problem as a Kalman filter where the state vector consists of deviations of O-D flows from prior estimates based on historical data. Based on the set of link counts obtained at the end of each interval, the O-D flow deviations predicted for that interval are modified. This process makes the use of information about travel times and path choice fractions for vehicles that are already on the network. Apart from generating estimates of O-D matrices for the current time interval, the model can also predict O-D matrices corresponding to future departure intervals and update O-D matrices corresponding to past departure intervals. Van Aerde's et al. (1993) QUEENSOD back-calculates origin/destination demand estimates from the dynamic link flows and speed measurements derived from dynamic link use probabilities from simulation. They consider that vehicle departures during one time slice are often only observed as flows on downstream links during several subsequent time slices, where this lag varies as a function of the travel time experienced during earlier portions of the trip. This effect is apparent during periods of congestion. when the trip makers require an increasingly larger number of time slices to complete their entire trip towards their final destination. Although this is not a strictly dynamic assignment-based approach, it performs the temporal and spatial mapping of link flows of the assignment-based approaches.

Sherali et al. (1997) consider the estimation of split parameters that prescribe an origin/destination trip table based on dynamic information regarding entering and exiting traffic volumes through an intersection or a small freeway section. They developed two models to solve this problem, one based on the Least Squares estimation approach and the other based on a Least Absolute Norm approach. Both synthetic as well as realistic simulated data was used. They used traffic counts at the entrance and exit at constant time intervals (referred to as sampling intervals or time periods) in order to estimate split parameter matrix $\left[\beta_{ij}\right]$ where, β_{ij} is the proportion of traffic entering entrance i that is estimated to leave the facility via exit j. These split parameters are then used to minimize the error between the observed and estimated flows. They used Least squares, where the squares of the errors are minimized and least absolute norm approach where the sum of absolute deviations are minimized.

Another way of classifying different approaches is to separate them according to the three main types of network. The first type is the linear network or freeway corridor, and in this case, the issue of route choice is not considered. The second type is intersection and involves the estimation of intersection turning fractions. The last type is the general network which is the most challenging because it captures both the temporal and the spatial aspects of the network. Approaches for linear corridor include Nihan and Davis (1987), Camus et al. (1994), Cremer and Keller (1987), van der Zijpp and Hammerslag (1994), and van der Zijpp (1997). Extensions for intersections include Nihan and Davis (1989). General and more complex networks are discussed by Ashok and Ben-Akiva (1993), Bell (1991), and Van Aerde (1993).

This research presents a practical approach for dynamic origin/destination demand estimation. The proposed dynamic origin/destination demand framework will address many of the shortcomings of existing formulations and present a formulation for general networks and not just corridors.

As was mentioned previously, most methods make the assumption that the time taken by vehicles to traverse the network is smaller than the algorithm step size; however, this constrains procedures for use in either very small networks or to have a time step that is too large to be effective. This problem is addressed by using a DTA (Dynamic Traffic Assignment) model formulated by Jayakrishnan et al. (1997). DTA captures the traffic dynamics by incorporating the temporal dimension into the traditional traffic assignment problem. The approach analytically incorporates a hydrodynamic traffic model as in a simulation into an optimization-based dynamic traffic assignment framework. This is the first such model where simulation equations are incorporated into as constraints in an optimization framework for network assignment. This results in a set of recursive equations used to compute the link travel time that guarantees the First-In-First-Out (FIFO) requirement without the need to explicitly impose FIFO constraints. Network assignment is accomplished for the time-dependent OD demands based on a two-level optimization framework which fixes the path-link incidence indicator between the time-dependent paths and their constituent links at one level and then assigns traffic similar to a static traffic assignment at the other level. The assignment is performed based on link loads (number of vehicles) as opposed to link flows used in static traffic assignment.

Using this analytically embedded dynamic traffic assignment framework, four formulations were developed by Jayakrishnan et al. The dynamic user equilibrium

(DUE) traffic assignment formulation is based on a dynamic generalization of Wardrop's first principle which requires for all used paths between an origin-destination pair departing within a given time interval to have equal minimal travel time and no unused paths can have a lower travel time. This formulation is important in the estimation and prediction of traffic states based on a behavioral route choice principle. The dynamic system optimal (DSO) traffic assignment formulation is based on Wardrop's second principle which aims to minimize the total travel time spent in the network. It uses marginal travel times rather than average travel times, as in the DUE problem, for route choice evaluation (i.e., equilibrating the used paths with marginal travel times). The difficulty of this problem is that the marginal path travel time found in the optimization must be consistent with the actual time use of each link along the path. This implies that not only the marginal travel time needs to be kept track of, but the actual travel time must be retained as well in order to compute the correct marginal travel time paths. Optimality conditions for the DSO problem have been derived and shown to be equivalent to the total travel time of the system being minimal and a solution procedure has been developed to solve the DSO problem.

The third formulation considers queuing on arterial streets. The arterial links are modeled as two sub-links: a moving link and a queuing link. This better captures the traffic phenomenon on arterial streets and also enables physical queue lengths to be obtained within the optimization framework. A solution procedure was developed by them that explicitly considers queue formation and dissipation in the arterial streets. The last formulation addresses on-line DTA implementation using rolling horizon procedure. The formulation of the on-line DTA model is significantly more complex due to the temporal dependencies of vehicle conservation across stages. This will require a path-based DTA formulation as well as a path-based solution procedure to handle the previously assigned vehicle packets that have not reached their destinations. Several dynamic rolling horizon assignments were formulated according to the capability to re-route the unfinished trips.

The result of using density with the DTA model allows capturing of density changes that would otherwise not be captured by using flow, since flow is considered to be steady for a longer time step. Among the three flow parameters (speed, flow, density), density is a direct measure of traffic demand. Density is not only a quantitative measure, but it is also a qualitative measure of the “closeness” of vehicles, which can reveal even the psychological comfort of drivers.

The common practice is to use traffic flow as a measure of traffic demand. However, demand does not occur as a rate of flow. Traffic demand is generated (or absorbed) at sources and sinks according to land use. The origin/destination demand results in the network loading of individual links. The vehicles on these links is what produces the rate of flow and the speed. Another problem with using flow is the fact that the relationship between flow and speed is not monotonic, thus the relationship between the flow and travel time (inverse of speed) is also not monotonic. This behavior of the speed-flow curve causes many problems in algorithm formulations that use flow as a variable and require the computation of travel time cost. On the other hand, density and speed (or travel time) possess a monotonic relationship. Travel time is important since it is a measure of the traffic system cost. The use of section density maintains a convex travel time cost function (Figure 2.1) which is important for convergence characteristics.

The fundamental difference between density and flow is that density is measured over a length of space at a particular instant in time, while flow is measured over a period of time at a particular point in space. In dynamic frameworks, density should clearly be the traffic variable of choice and not flow.

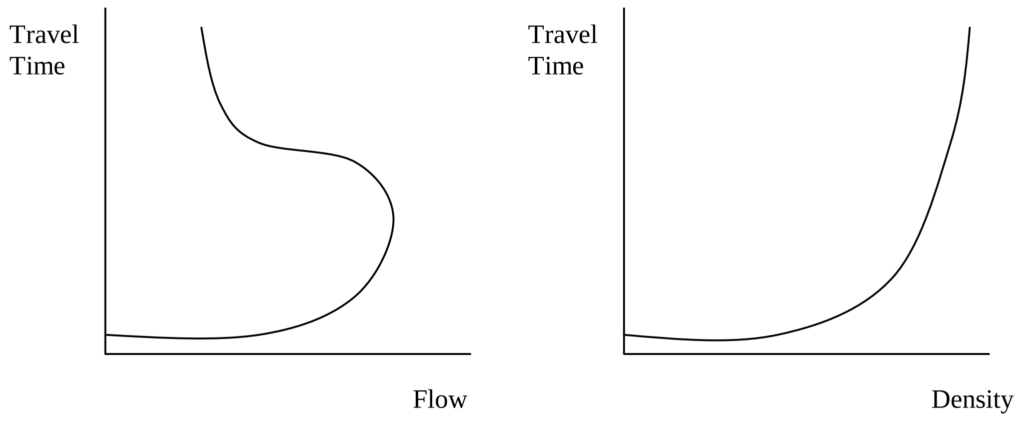


Figure 2.1 Travel Time cost function comparison

Jayakrishnan et al. (1994) developed DYNASMART (DYnamic Network Assignment Simulation Model for Advanced Road Telematics), an evaluation model that incorporated the driver response to information, the traffic flow behavior, and the resulting changes in the characteristics of network paths, into an integrated simulation framework. The model is based on simulating individual vehicle movements according to macroscopic flow principles, the driver path selection behavior under information being explicitly modeled. Detailed modeling of intersection delays as well as a variety of traffic control options for both freeways and arterials can be obtained from this model. The model has ability to model route choice of drivers with or without access to ATIS information and responsiveness to dynamic O-D information available to the controller as reported by the ATIS or other sources. It also has ability to predict the time-dependent impedance (travel time) based on the assignment results, and provide feedback to the control center that may be used in the assignment of vehicles.

The challenge in dealing with dynamic formulations is the inclusion of short time steps in addition to the usual spatial space patterns. Consequently, static methods that use flows aggregated over time are no longer valid. To determine the dynamic origin/destination demand matrix, the number of vehicles exiting from sources and entering into sinks during a small time period needs to be estimated. This demand results in a traffic pattern that is distributed onto the network. Therefore, the demand is what produces the traffic pattern, but the traffic pattern does not cause the demand even though it changes the route choice of the demand. Past formulations use vehicle flows (or even occupancy) as the input variable to both dynamic origin/destination demand estimation and traffic assignment formulations. However, the danger with this approach is the fact that a point measure such as flow, is assumed to be spread evenly across the link.

Sun (1998) has shown that point values are rather unstable and do not accurately reflect the true traffic distribution over the entire roadway section. He demonstrated the inadequacy of using local or point measures for estimating section travel times. Point travel times or point speeds fluctuates significantly and should not be extrapolated over the entire section. A more appropriate parameter to use in dynamic formulations is section density. By knowing the number of vehicles on all the links of a network at small time steps, one can construct the origin/destination demand pattern that produces such traffic distributions through time series tracking of the demand pattern.

This page is left blank intentionally.

CHAPTER 3. REAL DATA

3.1 INTRODUCTION TO CORBA (Common Object Request Broker Architecture)

In the early days of computing, computers operated independently of one another with no communication between them. The hardware and the system software were proprietary and supplied by a single vendor. The software applications were usually custom-developed for specific purposes. Data sharing between systems was minimal and done the old-fashioned way by physically transporting tapes or other storage media from one system to the other. The next step was to connect the computers and devices in a network using propriety protocols for data communications. This was followed by the development of standard protocols and open systems. The era of open systems also led to system integration in which the customers could choose various hardware components from different vendors and integrate them to create a custom configuration suiting their computing needs and cost requirements. Hardware itself was designed and built using off-the-shelf components: microprocessors, memory chips and other such building blocks.

Over the years, there has been a steady move in the direction of buying software off the shelf and customizing it. In fact, software costs as a percentage of total system costs are increasing when compared to hardware costs. However, the various software applications in a typical system generally do not communicate with each other, even if they do, such communication is minimal. Further, the cost of developing and maintaining software is skyrocketing. Techniques ranging from structured programming and modular programming to object-oriented programming have been proposed both to reduce software cost and to promote software reuse.

While object-oriented programming has resulted in considerable software reuse and cost reduction, it is still not enough. More work is needed to realize the ultimate dream of buying software components from various vendors, integrating them to form applications, and integrating these applications to form complete systems. There is a need for software components to be able to communicate with each other using “standard” mechanisms and “open” interfaces for an effective integration to occur. As software and hardware systems get more complex, the need for interoperability among different components becomes critical.

CORBA (Common Object Request Broker Architecture) was proposed as a standard by the Object Management Group (OMG), a consortium with over 700 members. Leading companies such as Netscape, IBM, Sun, Motorola, Nokia, Boeing, and others are using CORBA to build industrial-strength inter-operable software. By its very design, CORBA leads to the building of object-oriented, distributed applications. CORBA is a middle-ware that handles and manages all the transactions between a client and a server. It acts as a broker between the client and the server.

CORBA is a software middle-ware (or bus) that allows applications to communicate with one another, regardless of who designed them, the platform they are running on, the language they are written in, and where they are executing. CORBA also enables the building of plug-and-play components software environment.

The basic building block in CORBA is the CORBA object. CORBA objects are “blobs” of intelligence that can live anywhere on a network. They are packaged as binary components that remote clients can access via method invocations. It models a real world object and consists of data and methods that may be invoked on it. An object’s public interface is defined through the CORBA Interface Defining Language (IDL). CORBA IDL is a definition language that hides the underlying object implementation. A client of the object simply uses this interface to invoke methods—it does not know nor care about location, platform, or implementation details of the object. It can be in the same process or on a machine that sits across networks. In addition, clients don’t need to know how the server object is implemented. For example, a server object could be implemented as a set of C++ classes or it could be implemented with a million lines of existing COBOL code—the client doesn’t know the difference. What the client needs to know is the interface its server object publishes. This interface serves as a binding contract between clients and servers.

The Object Request Broker (ORB) is a middle-ware component that implements the CORBA bus and acts as a broker between the client and the server. The ORB manages the client request and hides the location and implementation details from the client. It lets objects discover each other at run time and invokes each other's services. It lets objects transparently make requests to—and receive responses from—other objects located locally or remotely. The ORB intercepts the calls and is responsible for finding the object that can implement the request, pass it the parameters, invoke its methods and returns the results. The client need not be aware of where the server object is located, its programming language, its operating system, or any other system aspects that are not a part of the object's interface.

The advantages of using CORBA are as follows:

1. It helps provide an open/standard interface between clients and servers.
2. It helps create an extensible application framework.
3. It helps create programming language independent applications.
4. It helps create very flexible web-based applications.
5. CORBA based applications have the capability of scalability.

3.2 CONCEPTUAL FRAMEWORK OF THE CORBA CLIENT/SERVER

A CORBA client was designed, developed and implemented in the TMC labs at UC, Irvine. This CORBA client successfully extracts the real-time loop detector data from various freeways in the Irvine network. There is a CORBA server running at the Caltrans District 12 (D12) office. This server stores all the real-time data from the controllers at various loop detector stations on freeways in the Irvine network. The framework of this CORBA server is as follows:

1. Obtains the real-time controller data and stores these as CORBA objects.
2. Provides the root handle of the application.
3. Also provides the individual handles to these CORBA objects.

The framework of the CORBA client running at the TMC lab in UC, Irvine is as follows:

1. Makes a connection to the D12 server to see if it is up and running.
2. Gets the root handle of the application.
3. Gets the individual handle to the requested CORBA objects.
4. Retrieves the list of requested CORBA objects.
5. Retrieves the raw data from these CORBA objects.
6. Resolves (processes) this raw data.
7. Displays the processed data.

The controller stores the vehicle flow and the detector occupancy for every thirty second period. This is the finest resolution of real-time data available in real-time from these controllers. This thirty second data comes in as raw data in the form of a binary string which is then processed before being displayed.

3.3 REAL-TIME LOOP DETECTOR DATA

Real-time traffic data (thirty second flows and occupancies) were collected on freeway 405 North at two loop-detector stations. The section of Irvine network chosen included 405 North, Jeffrey and Sand Canyon. 405 North has four lanes. Initially, various databases of twenty-four hours were prepared for seven consecutive days to study the relation between traffic pattern and the time of the day. It was observed that the traffic volume is at its lowest from 2:30am to 3:30am. During this period the thirty second traffic flow counts were typically in the range of one to three vehicles.

Data collected on March 04 and March 05, 1999 for a twenty-hour period is used to analyze the potential of using real data for the dynamic OD demand estimation purposes. A 24 hour plot (Figure 3.1 and Figure 3.2) of the real data (thirty second flow) from 3:45am on the March 04, '99 to the 3:45am on the March 05, '99 shows that the thirty second volumes at 3:45am are typically of the order of 2-3 vehicles and the increases as time progresses, reaching its maximum around 7:00am and then drops after 9:00am (the a.m. peak). The a.m. and the p.m. peak can be observed very clearly.

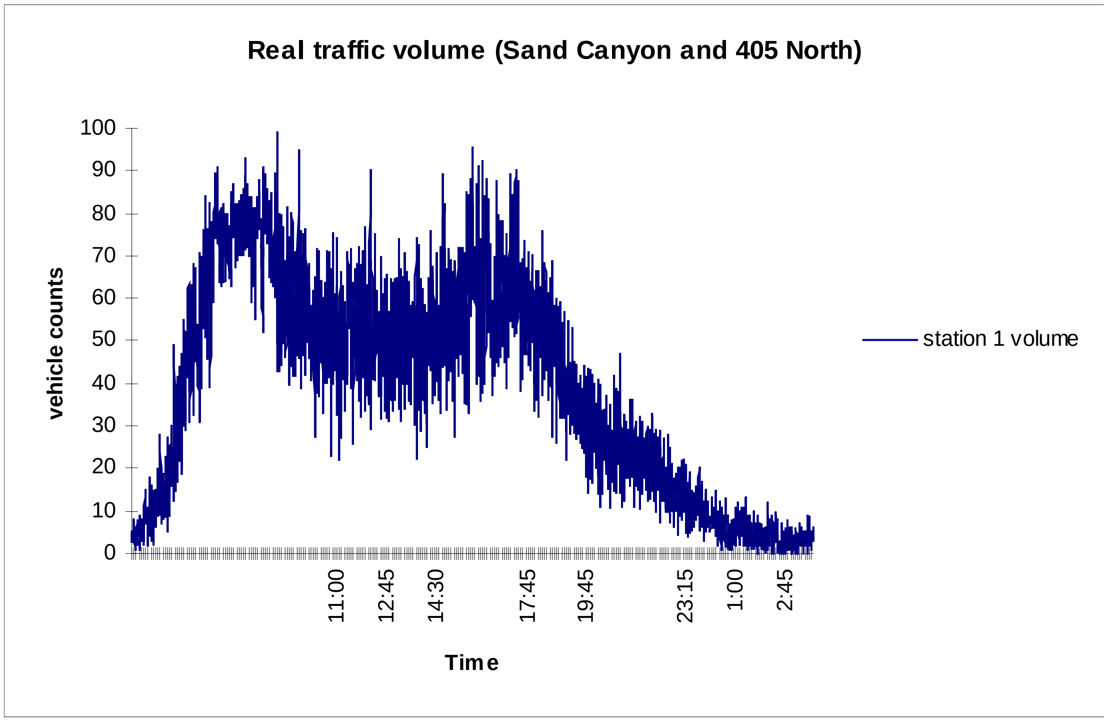


Figure 3.1 24 hour plot of real data (upstream station)

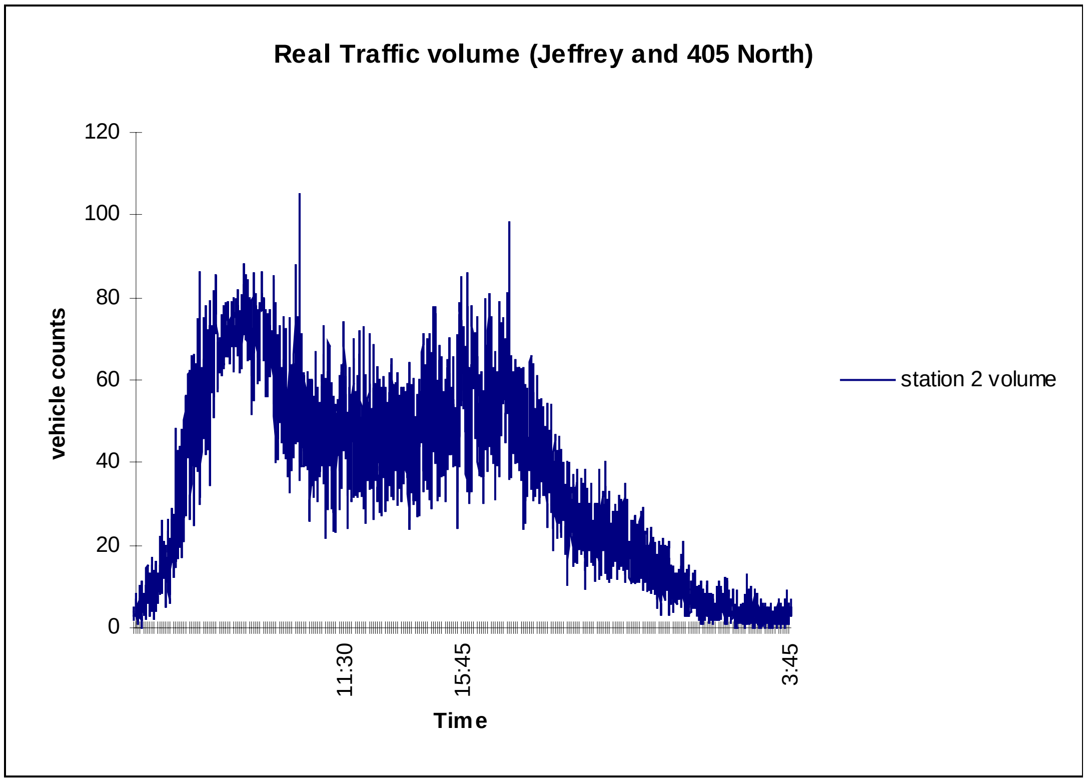


Figure 3.2 24 hour plot of real data (downstream station)

An important issue associated with the use of real data from loop detectors is that of the controller accuracy involved in counting the number of vehicles. Another issue that arises using in real data for dynamic OD demand estimation is in getting the real OD-link incidence matrix (the 'B' matrix, explained in Chapter 4). The real 'B' matrix can be obtained by putting the cameras at every intersection, using license plate re-identification or use of Automatic Vehicle Identification (AVI) and tracking individual vehicles. Tracking individual vehicles is a time consuming process and may involve certain types of mapping errors.

Section densities were derived on the above chosen section of Irvine network using the real data. The portion of the network between Sand Canyon and Jeffrey on 405 North has one on-ramp and one off-ramp (Figure 3.3). The data from these on and off ramps was not available. So, when we computed the section density using the section load formula (given below) there was an excess of vehicles being generated on the section. This implies that there is a positive net inflow of vehicles from the on and off ramps (on-ramp volume - off-ramp volume).

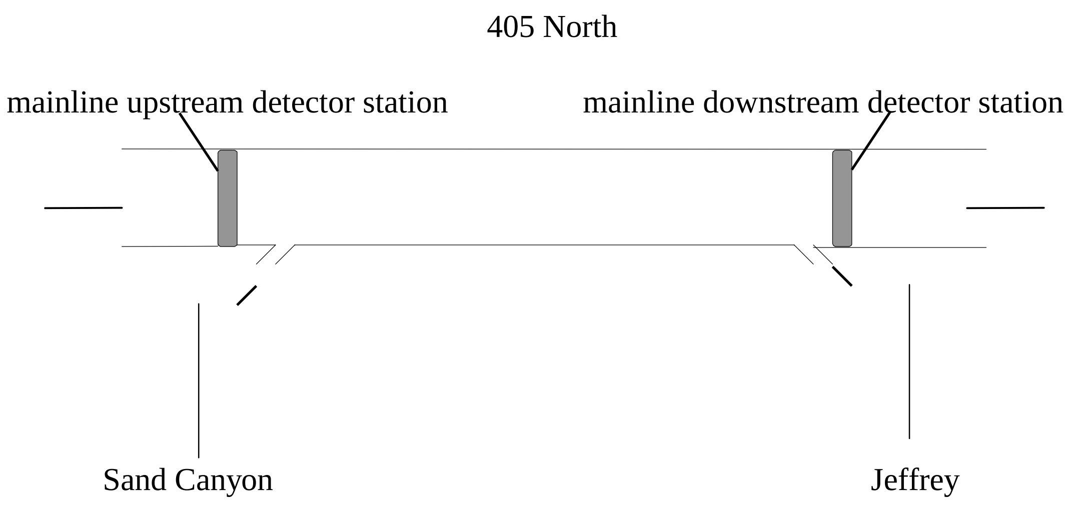

405 North
mainline upstream detector station
mainline downstream detector station
Sand Canyon
Jeffrey

Figure 3.3 Section of Irvine network used (showing on and off-ramps)

Mathematically section density is expressed as

$$L^t = L^{t-1} + \frac{\Delta t}{D}\left(V_u^t - V_d^t - E\right)$$

where,

V_u^t = Volume at the upstream station at time t.

V_d^t = Volume at the downstream station at time t.

L^t = Section density at time t.

D = Length of the section

Δt = Time interval

$E = V_d^t - V_u^{t-lag}$, where lag = the average travel time in numbers of time intervals.

Or in other words, E = the estimated net flow from ramps i.e., off-ramp volume minus on-ramp volume over lag+1 time intervals.

Section density is defined as the number of vehicles per mile on the section of the link at a particular instant of time. The length of the sample section, D, is 0.82 miles. The time interval used was 30 seconds and the lag was 6 time intervals.

The density curve over the 24 hour period is shown in Figure 3.4. The a.m. and the p.m. peaks are very clearly visible. Section density shows less variability when compared with flow over the same time intervals. The maximum observed density was 140 vehicles per mile per lane.

In order to reflect the net flow from the off-ramp and on-ramp in the sample section, we assumed that the mainline vehicle platoon travels at an average speed. This is not an overly restricted assumption for a sample section of this length. From this average speed, we obtain an average platoon travel time for this section. This travel time is the number of lag intervals that is used to compute the net flow from the ramps over the lag+1 time intervals.

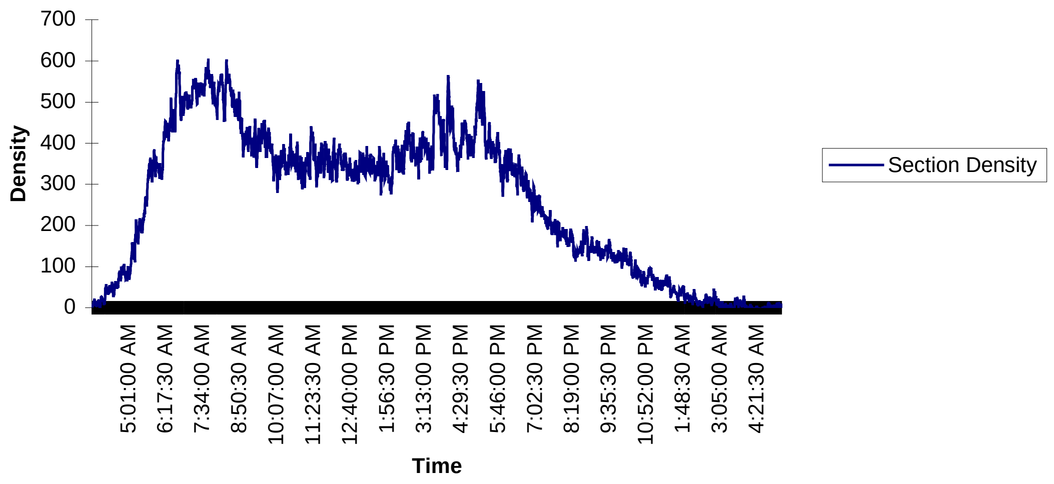


Figure 3.4 24 hour plot of section density

The computed densities over this sample section serves as an example of how section densities can be derived for use in dynamic OD estimation. This demonstrates the potential of using densities instead of merely relying on flows for dynamic formulations.

CHAPTER 4. DYNAMIC OD ESTIMATION USING DTA

4.1 INTRODUCTION TO DTA

The Dynamic Traffic Assignment (DTA) model that is proposed for use in the estimation of origin/destination demand matrix is a bi-level optimization framework that incorporates fundamental traffic flow relationships. The use of traffic flow relationships is significant since it bypasses the necessity to use a flow dependent link cost function such as the well-known BPR (Bureau of Public Roads) link cost function. The bi-level program consists of a upper problem that seeks to find the minimum time dependent travel times to all the nodes in the network assuming the equilibrium traffic loads on the links to be fixed by a previous iteration of the lower problem. And the lower problem is a assignment process using the nodal arrival estimates from the upper problem. By iterating between the two problems, a dynamic assignment is obtained that incorporates the temporal element of traffic arrival and departure.

The origin/destination demand matrix estimation and the traffic assignment problems are mutually dependent and complimentary. Traditionally, in static planning applications, the OD demand is estimated first, and then the demand is distributed to the network according to an assignment algorithm. However, in a dynamic framework there needs to be constant iterating between the OD matrix estimation module and the traffic assignment module in order to capture the system dynamics.

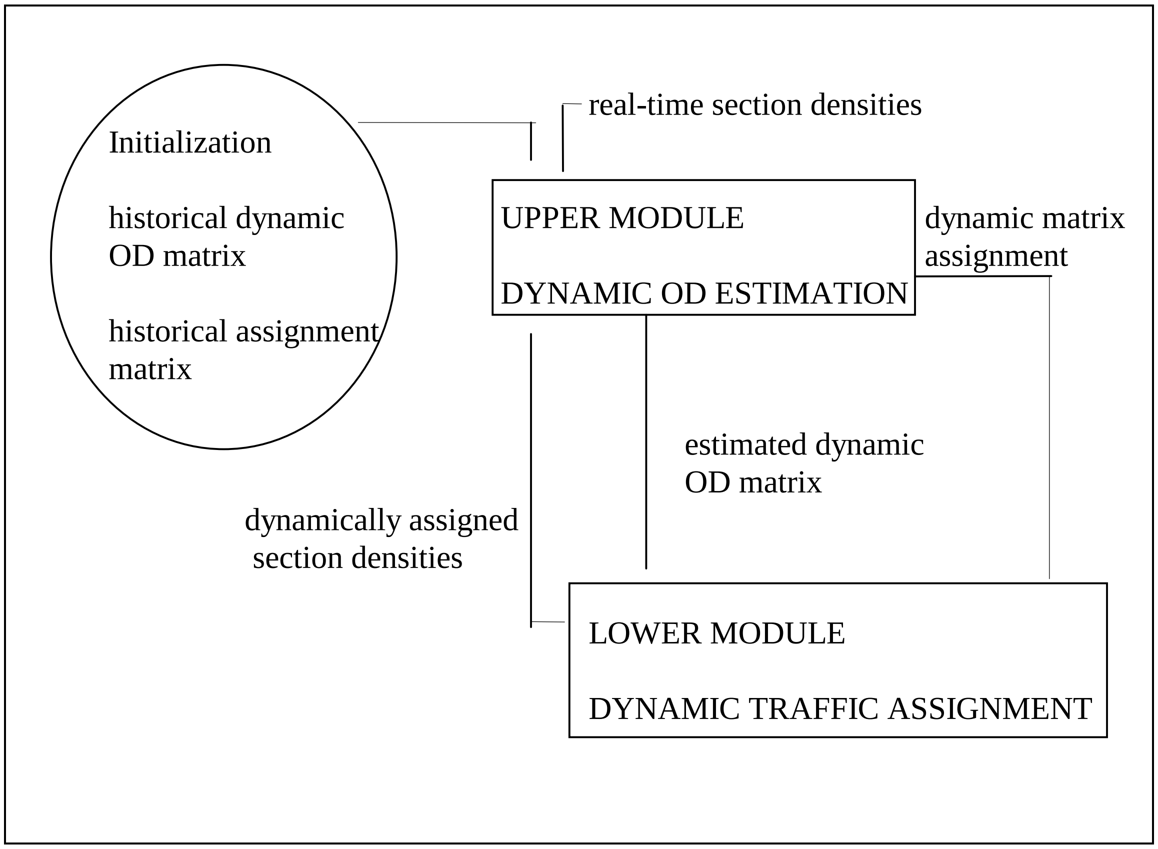

Initialization
historical dynamic OD matrix
historical assignment matrix
real-time section densities
UPPER MODULE
DYNAMIC OD ESTIMATION
dynamic matrix assignment
estimated dynamic OD matrix
dynamically assigned section densities
LOWER MODULE
DYNAMIC TRAFFIC ASSIGNMENT

Figure 4.1 Dynamic OD Demand Matrix estimation Framework

Figure 4.1 gives a graphical representation of the iterative dynamic OD matrix estimation/traffic assignment framework. The initialization can require both historical OD matrices and initial assignment matrices (or route proportions). There are different sources for such information, and the selection of ideal initial conditions need to be investigated. For example, if an OD seed matrix is needed, then it can come from possibly a time-of-day historical OD matrix or from previous time steps. An initial assignment matrix can also be obtained by several methods. One method is to use a historical assignment matrix. Another is to use time proportions derived from shortest path calculations.

The ODE (Origin/Destination demand matrix Estimation) will estimate the OD matrix that generated observed section densities. The DTA (Dynamic Traffic Assignment) in turn uses these demands and returns to the ODE newly computed section densities and a dynamic assignment matrix. The equilibration between the two modules will minimize the discrepancies between observed section densities and the ones computed by the framework.

Initially, such a framework will be created off-line. This will allow a concentrated effort on the formulation of the problem instead of an overbearing concern over the computational requirements. Once an off-line framework is proved to be robust, it can be extended to the real-time case. The feasibility of the computational burden has already been investigated to some extent.

Several indices are needed in order to describe the temporal and spatial components of variables. In order to clarify our notation, the following summary is presented:

y	section density
$y(k,m)$	section density for the k^{th} time step on link m.
q	traffic demand
$q(i,l)$	origin demand from i originating at time l.

b	origin/destination demand-link incidence matrix components
$b(i,j,k,l,m)$	proportion of the demand from origin destination pair ij originating at time l that is on link m at the current time step k
f	path flows that implicitly contribute to the link densities
$f(i,j,l,p)$	path flows from a route p used by origin/destination ij that originated at time l
δ	unit impulse or sample
$\delta(i,k,l,m)$	unit impulse=1 if k is the estimated equilibrium arrival time at link m for a vehicle originating from i in time step l, else impulse=0
λ	travel time
$\lambda(i,l,n)$	travel time from origin i leaving at time l to node n
N	set of items
N_{ij}	set of origin/destination pairs
N_p	set of paths belonging to a particular origin/destination pair
N_i	set of origins
N_l	set of originating times
N_n	set of nodes in the network
N_m	set of links
N_k	set of time steps
L	length
T	time step

$\#\left(N_{ij}\right)$	number of origin/destination pairs

Indices:

i	origin
j	destination
k	current time step
l	originating time step
m	link
p	designates the specific route used in an origin/destination commute
n	node

In matrix notation, the relationship between observed link densities y and the demands q is simply described as:

$$y = B.q$$

Here, the spatial-temporal assignment matrix B, maps demands from all origin/destination pairs over all generating time periods to all links over all travel time periods. The matrix B is then the origin/destination-link incidence matrix.

Each section density is derived from the originating demands as follows:

$$y(k,m) = \sum_{l=k-N_t T}^{k} \sum_{i,j=1} b(i,j,k,l,m)q(i,l)$$

Here, N_t is the maximum number of time steps required for a vehicle from any origin to arrive at link m so l represents only the times where an origin with a relevant demand is generated. Also, N_{ij} is the total number of origin/destination pairs.

The above equation shows how all the demands from all origin/destinations that originated from all relevant previous time steps are mapped to link m at the current time k via the origin/destination-link proportion $b(i, j, k, l, m)$. It can be seen that the most complicated component of this expression is the origin/destination-link proportions which captures the effects of route choice, travel time, dispersion, and departure time. It is a complete description of the temporal-spatial characteristic of the traffic on a network.

In order to obtain the origin/destination-link incidence matrix, B, the DTA (Jayakrishnan et al, 1995) model with traffic flow relationships is used in this formulation.

The DTA model uses the following bi-level programming approach:
Upper Level: Time dependent shortest path problem to determine travel distances of vehicles based on traffic assignment by the lower problem and the estimated equilibrium arrival times.
Lower Problem: Augmented static assignment problem that uses virtual paths resulting from the incorporation of the time dimension.

The time dependent shortest path problem is formulated as a dual problem that seeks to maximize the sum of the travel times to every node from every origin for all originating time steps. This is expressed as follows:

$$\text{max. } Z_U = \text{max.} \sum_{i \in N_i} \sum_{l \in N_l} \sum_{n \in N_n} \lambda(i,\ l,\ n)$$

The objective function is constrained by the following arc constraints that resemble the minimum path cost requirement of the standard shortest path algorithms:

$$\lambda(i,\ l,\ n)\delta(i,\ k,\ l,\ n) \ \le\ \left[\lambda\left(i,\ l,\ n_{-1}\right) + t\left(k_{-1},\ m\right)\right]\delta\left(i,\ k_{-1},\ l,\ n_{-1}\right)$$

$$\forall \quad i \in N_i, \quad n, n_{-1} \in N_n, \quad m \in N_m, \quad k \ge k_{-1} \in N_k, \quad l \in N_l$$

Here, n_{-1} denotes the head-note of link m which has an endnode of n, and k_{-1} denotes a time prior to the current time of arrival at node n of k. This constraint states that the shortest path travel times from an origin i to a node n is always less than the path travel time from an origin i to a node n_{-1} plus the link travel time of any link that end in node n. This constraint effectively guarantees a shortest path tree in both the space and time dimensions.

The following constraint on the equilibrium arrival times is also needed:

$$\sum \delta(i,\ k,\ l,\ n) \ = \ 1 \qquad \forall \quad i \ \in \ N_i, \quad l \ \in \ N_l, \quad n \ \in \ N_n$$

This constraint states that the unit impulse representing the travel time of a vehicle originating from i in time l can only arrive at node n during one and only one time k. This constraint is necessary because of the use of a dynamic link-path incidence matrix instead of an explicitly equilibrium nodal arrival variable.

The conventional static user-equilibrium is augmented with the time dimension via dynamic links and paths and the tracking of vehicle traversal distances. The use of a path-based assignment algorithm works well in this instance since in contrast to the static problem, the flow is not assumed to exist uniformly throughout a path. In fact, flows are altogether replaced by section densities. Also, path-based assignment leads to a simple derivation of the origin/destination-link incidence matrix since temporal path flow values are kept automatically. The dynamic user-equilibrium assignment objective function then becomes:

$$\text{min.} \quad Z_L \ = \ \text{min.} \quad \sum_{k \in N_k} \sum_{m \in N_m} \int_0^{y(k,m)} t(m,\ L(k,\ m),\ \omega) d\omega$$

Here, $t(m,\ L(k,\ m),\ \omega)$ is the dynamic cost (travel cost) on a link m, with a section density of ω, and a vehicle traversal distance $L(k,\ m)$. This expression differs from the static version in the use of a time dependent link density and a time dependent cost function.

The following standard conservation, non-negativity, and definition constraints also apply here:

$$q(i,\ j,\ l)\ =\ \sum_{p \in N_p} f(i,\ j,\ l,\ p) \qquad \forall i,\ j$$

The conservation equation specifies that all path flows for an origin/destination pair originating from the same time must be conserved.

$$f(i,\ j,\ l,\ p)\ \geq\ 0$$

The non-negativity constraint requires that no path flow can ever be negative, and this insures the non-negativity of section densities also. This is necessary since negative flows do not reflect real traffic conditions.

$$y(m,\ k)\ =\ \sum_{i,\, j \in N_{ij}} \sum_{l \in N_l} \sum_{p \in N_p} f(i,\ j,\ l,\ p)\delta(m,\ p,\ k)$$

This definition constraint simply expresses the section density as the sum of all dynamic path flows that traverse that section at a certain time.

$$q(i,\ l,\ n)\ =\ \sum_{d \geq l} \Big(\sum_{m_0 \in N_0(n)} y(i,\ k,\ d\ +\ 1,\ m_0)\delta(i,\ k,\ d\ +\ 1,\ n)\ -$$

$$\sum_{m_i \in Ni(n)} y(i,\ k,\ d,\ m_i)\delta(i,\ k,\ d,\ n)\Big)$$

$$\forall \quad i\ \in\ N_i, \quad l\ \in\ N_l, \quad n\ \in\ N_n$$

The additional Kirchhoff's constraint specifies that the difference between the flow that enter a node *n* and exit a node *n* equals the flow generated at origin *n* in time *l*. Here, m_0 and m_1 designate the links that enter and exit node *n*; *y(i,k,l,m)* specifies the section density with flow from origin *i* originating at time *l*, and is on link *m* at time k; and *δ(i,k,d,n)* is an unit impulse with value 1 if *k* is the estimated equilibrium arrival time at node *n* for a vehicle originating from *i* in time step *l*.

Also, the section density is constrained to have a value less than jam density as follows in order to properly capture shock wave propagation:

$$y(k, m) \leq y_j(k, m) \qquad \forall \quad m \in N_m, k \in N_k$$

After convergence, the origin/destination-link incidence proportions are derived from the resulting path flows of the lower level problem. In other words,

$$b(i, j, k, l, m) = \sum_{p=1}^{N} \frac{f(i, j, l, p)\delta(m, p, k)}{q(i, j, l)}$$

Here, N_p is the number of paths for a particular origin/destination pair ij, and $\delta(m, p, k)$ is a unit impulse and has a value of one when link m is used in path p during time k. So, the sum of the ratios of the path demands over the origin/destination demand will give the ratio of the section demand over the origin/destination demand.

4.2 CONCEPTS IN KALMAN FILTER

One formulation of dynamic Origin-destination Demand estimation framework uses the Kalman filtering technique. References on Kalman filtering abound including Santina (1994). Kalman filtering is an optimal state estimation process applied to a dynamic system that involves random perturbations. More precisely, Kalman filter gives a linear, unbiased, and minimum-error variance recursive algorithm to optimally estimate the unknown state of a dynamic system from noisy data taken at discrete real-time. It is a widely applied method for parameter estimation in dynamic systems. It has been used in many areas of tracking systems, satellite navigation, missile trajectory estimation and radar control, to name a few. With the recent development of high speed computers, the Kalman filter has become more useful even for very complicated real-life applications. However, the use of Kalman Filter for dynamic origin/destination demand estimation requires rolling horizon time frame measurements and updates in real world operations. Testing this is beyond the scope of this final report; however, a one step version of the Kalman Filter is tested to check its efficacy.

Kalman filter is chosen for the following reasons:

1. It is suitable for linear systems. The Origin Demand matrix estimation problem can be formulated with a linear mapping from the state variable to the measurement variable.
2. The formulation is stochastic which allows for measurement and state errors, and the initial state could be described stochastically.

3. It is dynamic and hence suitable for OD matrix tracking over time.
4. The formulation is recursive and allows for each additional measurement to be incorporated sequentially. With previous estimate and the new measurement, a new estimate is produced.

Before a Kalman filter can be used two equations must be supplied: the state equation and the measurement equation. The state equation describes how the unknown parameters evolve through time. The measurement equation describes the relation between the unknown parameters and the measurements. In both equations it is possible to specify uncertainties by way of noise terms.

The Kalman filter tries to produce estimates which are linearly related to the measurements such that the expected sum of squares is minimized. In other words, it seeks to minimize

$$E\left[(x - \hat{x}) \,/\, (x - \hat{x})\right]$$

where, x is the actual state variable,

$\hat{x}$ is the estimate.

Since, we assume the estimate of the state variable $\hat{x}$ to be unbiased, the expected value of the estimation error is zero. Therefore, the mean square error becomes the sum of the variances of the individual components in estimator error. Here, the state variable is the variable of interest, or origin/destination demands.

The basic equations of the Kalman filter are now presented. As in common practice, the estimate of a quantity such as x is denoted by a “hat” over the symbol for the quantity (i.e., $\hat{x}$). The argument for all estimates are written in the form ($i|k$), where the first index i is the step of the quantity estimated, and the second index k is the index of the most recent measurement used in making the estimate. The first measurement is assumed to be $z(1)$. Following are some of the notations used in the basic equations of the Kalman filter:

$\hat{x}(k+1|k)$: estimate of $x(k+1)$ based on $z(1)$, $z(2)$, $z(3)$, . . . , $z(k)$

$\hat{x}(k+1|k+1)$: estimate of $x(k+1)$ based on $z(1)$, $z(2)$, $z(3)$, . . . , $z(k)$, $z(k+1)$

$\hat{z}(k+1|k)$: is the estimate of $z(k+1)$ based on $z(1)$, $z(2)$, $z(3)$, . . . , $z(k)$

$\Delta x(k+1|k) = x(k+1) - \hat{x}(k+1|k)$ is the state prediction error.

$\Delta x(k+1|k+1) = x(k+1) - \hat{x}(k+1|k+1)$ is the state estimation error.

$\Delta z(k+1|k) = z(k+1) - \hat{z}(k+1|k)$ is the measurement prediction error.

$P(k+1|k) = E[\Delta x(k+1|k)\Delta x(k+1|k)]$ is the state prediction error covariance.

$P(k+1|k+1) = E[\Delta x(k+1|k+1)\Delta x(k+1|k+1)]$ is the state estimation error covariance.

The basic equations of Kalman filter are as follows:

Plant Model

$$x(k+1) = F(k)x(k) + w(k)$$

$$z(k+1) = H(k+1)x(k+1) + v(k+1)$$

Prediction

$$\hat{x}(k+1|k) = F(k)\hat{x}(k|k)$$

$$\hat{z}(k+1|k) = H(k+1)\hat{x}(k+1|k)$$

Correction

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K(k+1)\Delta z(k+1|k)$$

$$\Delta z(k+1|k) = z(k+1) - \hat{z}(k+1|k)$$

Kalman Filter Gain

$$K(k+1) = P(k+1|k)H'(k+1)\left[H(k+1)P(k+1|k)H'(k+1) + R(k+1)\right]^{-1}$$

Covariances

$$P(k+1|k) = F(k)P(k|k)F'(k) + Q(k)$$

$$P(k+1|k+1) = \left[I - K(k+1)H(k+1)\right]P(k+1|k)$$

where, $F(k)$ is the state coupling matrix

$H(k)$ is the output coupling matrix.

4.3 PROBLEM FORMULATION FOR DYNAMIC OD ESTIMATION

The state equation of the model in matrix notation is

$$q(k+1) = \sum_{h \in N_h} F(h)q(h) + w(k)$$

$$E[w(k)] = 0$$

$$E[w(i)w(k)] = \begin{cases} 0, & i \neq k \\ Q(k), & i = k \end{cases}$$

The state variable $q(k+1)$ denotes the vector of OD demands for every OD pair that started their commute at time $k+1$. The index h is a temporal index that represents every time step that is relevant to the current time step. In other words, the number of time steps of h includes all the previous time step demands that leads to the construction of current traffic pattern at $k+1$. Usually, h is a few sequential time steps previous to $k+1$, i.e., $k, k-1, k-2, etc...$ The methods for calculating an effective number of previous time steps, h need to be further investigated. An initial estimate of the length of all previous time steps, h, can be the average travel time of a vehicle to cross the entire network. The matrix $F(h)$ contains all the regression coefficients that relate the current OD demands to previous OD demands and also expresses the relationships across OD pairs. The error vector $w(k)$ is assumed to be white noise with zero mean, and $Q(k)$ is a $\#(N_{ij}) \times \#(N_{ij})$ symmetric and positive semi-definite covariance matrix. Whiteness implies that $w(k)$ is uncorrelated with itself. This equation expresses the auto-regressive nature of OD demand and assumes that the current OD demand can be related to the previous time OD demands. This is also a general formulation that allows for cross dependencies among the different OD pairs. Consequently, both the temporal and the spatial aspects of the network demands are captured with this formulation. For specific cases of networks, this formulation can be simplified for ease of computations. For example, in a corridor network, certain OD pairs do not share paths and their regression coefficient can be ignored.

The computation of $F(h)$ is no trivial task and requires careful investigation. The most straightforward way is to use linear regression to derive $F(h)$ for both the relationships across time and across space. However, this depends on the availability of the historical data for actual examples. If insufficient historical data is available, then some assumptions would need to be made in regards to the structure of the auto-regressive process being constant over time. Another simplifying assumption could be made which is to ignore the cross dependencies among OD pairs. This assumption may work better on sparse linear networks. than dense grid networks. This assumption would lead to a diagonal $F(h)$ matrix and consequently a diagonal $Q(k)$ matrix.

The measurement equation is

$$y(k+1) = \sum_{h \in N_n} B(k+1)q(k+1) + v(k+1)$$

$$E[v(k)] = 0$$

$$E[v(i)v(k)] = \begin{cases} 0, & i \neq k \\ R(k), & i = k \end{cases}$$

$$E[v(i)w(k)] = 0$$

The equation relates vector $y(k)$, a $1 \times l$ vector composed of the section densities of all the links in the network, with the origin/destination demands. Here, $v(k+1)$ is white noise with zero mean and known covariance $R(k)$ and is uncorrelated with $w(k)$. The matrix $B(k+1)$ maps the state variables to the measurements.

Initially, the assumption is made of a fully instrumented network where the section density of every link can be obtained. However, this scenario is not a requirement, and this scenario might not be realistic either, since perfect instrumentation is difficult to achieve. In real networks, there can be both recurrent and non-recurrent detector failures in addition to the fact that some segments are simply not instrumented. For example, the paving of roads would lead to the replacement of loop detectors on that road segment. In video detection, there can be transition periods such as dusk or dawn, where the detector is temporarily disabled for self-calibration purposes. In terms of non-recurrent failures, loops can experience such malfunctions as cross-talk, being stuck, or reduction of baseline inductance.

One of the most difficult values to obtain from the measurement equation is the OD/link incidence matrix, denoted by $B(h)$. All the previous dynamic OD matrix estimation schemes which require this matrix have always assumed that this variable was available, and issue of deriving such a variable has always been circumvented. This is however, not a correct approach, as the B matrix depends on the assignment of the assumed/estimated demands between the OD pairs. But now, by integrating a viable DTA module and a dynamic OD module this problem has been solved.

There is also an assumption that the initial state vector is probabilistic with zero mean, known positive definite covariance matrix $P(0)$ and is uncorrelated with both noise inputs, $w(k)$ and $v(k)$. In other words,

$$E[q(0)] = 0$$

$$E[q(0)q'(0)] = P_0$$

$$E[q(0)w'(k)] = 0$$

$$E[q(0)v'(k)] = 0$$

The notation $(k + 1 / k)$ is used to denote the estimate at time step $k + 1$ using up to the most recent measurement k. The Kalman gain can be computed as

$$K(k + 1) = P(k + 1 / k)B'(k + 1)[B(k + 1)P(k + 1 / k)B'(k + 1) + R(k + 1)]^{-1}$$

The Kalman gain is the expression that will minimize the mean square estimation error variances. The Kalman gain expression can be derived by a completing-the-square approach and also a differential calculus approach.

And by using the Kalman gain, the state vector can be estimated as

$$\hat{q}(k+1/k+1) = \hat{q}(k+1/k) + K(k+1)\left(y(k+1) - \hat{y}(k+1/k)\right)$$

In terms of optimization terminology, the Kalman gain multiplied by the section density error can be considered as the step size and direction of a move from the state vector from a previous time step to that in the current time step.

It may be expected that this model will not give as good results as the Least Square estimation presented next, as Least Squares is a complete maximum likelihood estimator and minimum mean square estimator of the state variables assuming auto-correlation, while Least Squares does not assume auto-correlation. However, this model is tested here to see if the results are substantially worse than the Least Square, since the rolling horizon version in the future will be a full Kalman Filter of which 1-Step version is a special case.

4.4 LEAST SQUARES FORMULATION FOR DYNAMIC OD ESTIMATION

Another formulation of the dynamic origin/destination demand estimation uses the least squares method. The least squares estimation procedure is used for determining an optimal estimate of origin/destination demands based on a set of noisy but unknown measurements of the section density. The section densities and the origin/destination demands are assumed to be linearly related. The recursive formulation allows additional measurements to be incorporated sequentially. Thus the new estimate is based on previous measurements and one additional new measurement. In other words, the least squares estimate based on $k + 1$ measurements is composed of a linear transformation of the least squares estimate based on the first k measurements plus a linear correction term based on the $k + 1^{th}$ measurement. The recursive nature of this procedure lends itself to a real-time application of the dynamic origin/destination demand estimation algorithm.

As in the Kalman filter estimation scheme, the least squares formulation also relates the origin/destination demands and the section densities via the following equation:

$$y = Bq + v$$

The basic non-recursive least squares seeks to minimize the sum of squares errors between the actual section density and the estimated measurements $B\hat{q}$

$$\text{min. } Z = \text{min. } \left(y - B\hat{q}\right)' \left(y - B\hat{q}\right)$$

where ’ stands for transpose.

The resulting origin/destination demand is expressed as

$$\hat{q} = (B'B)^{-1}B'z$$

The recursive least squares computation procedure allows the incorporation of the additional current measurement of section density. The procedure is as follows:

An initial estimate of the origin/destination demand q and the variable P (which we define later) is made. There can be different ways of doing this, but a simple method is to base it on historical values of the origin/destination demand and the origin/destination-link matrix B.

$$P(0) = \left[B_h' B_h\right]^{-1}$$

$$\hat{q}(0) = P(0) B_h' y_h$$

Here, the subscript h denotes that the values are computed from historical data. At each step a corrector gain $\kappa(k + 1)$ is computed

$$\kappa(k + 1) = P(k) b(k + 1)\left[b'(k + 1) P(k) b(k + 1) + 1\right]^{-1}$$

Also, $P(k + 1)$ is also computed for use in the next step

$$P(k + 1) = \left[I - \kappa(k + 1) b'(k + 1)\right] P(k)$$

The new values of the origin/destination demands are estimated using the corrector gain

$$\hat{q}(k + 1) = \hat{q}(k) + \kappa(k + 1)\left[y_{k-1} - b'(k + 1)\hat{q}(k)\right]$$

4.5 RESULTS USING KALMAN FILTER FOR DYNAMIC ODE

One network that was used was composed of six links and four nodes. The network is shown in Figure 4.2. The link characteristics are shown in Table 4.1. These characteristics are used in the computation of the link cost using a modified Greenshield’s equation in the upper and lower problems of the dynamic assignment module. There are two OD pairs. OD pair 1 has node 1 as the origin and node 4 as the destination. OD pair 2 has node 1 as the origin and node 3 as the destination.

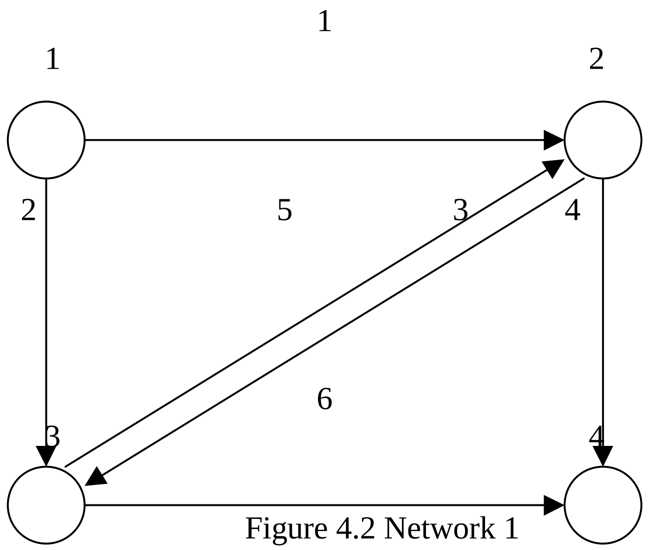


Figure 4.2 Network 1

Table 4.1 Link Characteristics (network 1)

Link	Jam Density (veh/mile)	Link Length (feet)	Max. Speed (feet/sec.)
1	160.0	2000.0	51.33
2	160.0	1000.0	51.33
3	160.0	750.0	51.33
4	160.0	2000.0	51.33
5	160.0	750.0	51.33
6	160.0	2000.0	51.33

Using the Dynamic Traffic Assignment module, the dynamic OD-link incidence matrix and the ideal link densities are obtained.

Three demand intervals are used in each trial run. The demand interval is larger than the one used for link densities because the demand on a single demand interval is translated to multiple link intervals. The assumption that the link time intervals are the same is restrictive. In the first example, the demand interval is 20 seconds, while the link interval is 10 seconds. The demand modeled for one minute is spread to 14 link intervals or two minute and twenty seconds (140 seconds).

Table 3.2 has four columns: true OD demand matrix, an arbitrary historical OD demand matrix used to initialize the dynamic OD estimation algorithm, the resulting estimated OD demand matrix, and the percent error from the true OD demand matrix. Eight trial runs of increasing demand (roughly increasing 20% for each successive run) are shown. For each run six rows are shown in the table. The six rows are the various OD pairs: row 1 is OD pair 1 at time step 1, row 2 is OD pair 2 at time step 1, row 3 is OD pair 1 at time step 2, row 4 is OD pair 2 at time step 2 and so on. Historical OD demand values were obtained by perturbing the seed demand values by an average of 12 percent for each trial run.

Table 4.2 Estimation results (Kalman Filter, network 1)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
5	5	0%	4.97	0.6%
10	9	10%	10.1	1.0%
5	4	20%	4.99	0.2%
7	6	14.3%	6.89	1.5%
9	8	11.1%	9.1	1.1%
6	5	16.7%	5.97	0.5%
<u>Run 2</u>				
5	8	60%	5.28	5.6%
10	8	20%	9.81	1.9%
5	8	60%	5.22	4.4%
7	8	14.3%	7.13	1.86%
9	8	11.1%	8.9	1.1%
6	8	33.3%	5.82	3%
<u>Run 3</u>				
13	12	7.7%	12.1	6.9%
12	11	8.5%	11.5	4.2%
11	10	9.09%	10.4	5.5%
10	9	10%	9.7	3%
11	10	9.09%	10.2	7.3%
12	11	8.5%	10.9	9.2%
<u>Run 4</u>				
20	23	15%	21.2	6%
30	32	6.67%	31.6	5.3%
20	22	10%	21.4	7%
25	22	12%	23.1	7.6%
15	18	20%	16.5	10%
20	18	10%	18.4	8%
<u>Run 5</u>				
25	30	20%	26.9	7.6%
32	35	9.3%	34.2	6.9%
25	30	20%	27.8	11.2%
30	35	16.67%	31.9	6.3%
30	35	16.67%	32.2	7.3%
25	28	12%	26.8	7.2%
<u>Run 6</u>				
30	35	16.67%	27.4	8.7%

35	40	14.2%	37.8	8.0%
30	35	16.67%	27.3	9.0%
35	40	14.2%	38.0	8.6%
38	45	18.4%	35.1	7.6%
30	36	20%	32.4	8.0%
<u>Run 7</u>				
40	45	12.5%	43.2	8.0%
45	50	11.1%	49.6	10.2%
42	48	14.2%	45.2	7.6%
44	50	13.7%	48.1	9.3%
48	55	14.6%	44.2	7.9%
46	52	13.1%	49.9	8.5%
<u>Run 8</u>				
50	58	16%	54.2	8.4%
58	65	12.1%	63.2	9.5%
60	65	8.3%	54.9	8.5%
55	62	12.7%	59.4	8.0%
62	65	4.8%	56.8	8.4%
58	65	12.1%	64.0	10.3%

It can be seen that the predicted OD demand values are very close to the true OD demand values when the true OD values are small and the historical OD values are close to the true OD demand values. The deviation of estimated OD demand values from the true OD demand values increases for the higher values of true OD (Figure 4.2). This is because of the fact that higher origin/destination demand results in a higher loading of traffic onto the network and this may result in traffic congestion which makes it difficult to map the section measures to the origin demands and hence, to estimate the OD demand matrix.

The average percentage errors are also higher when the historical (seed) OD demand values were farther away from the true OD demand values. This is due to the fact that it becomes difficult to converge to the true origin/destination demand values if the seed OD matrix is farther from the true OD matrix.

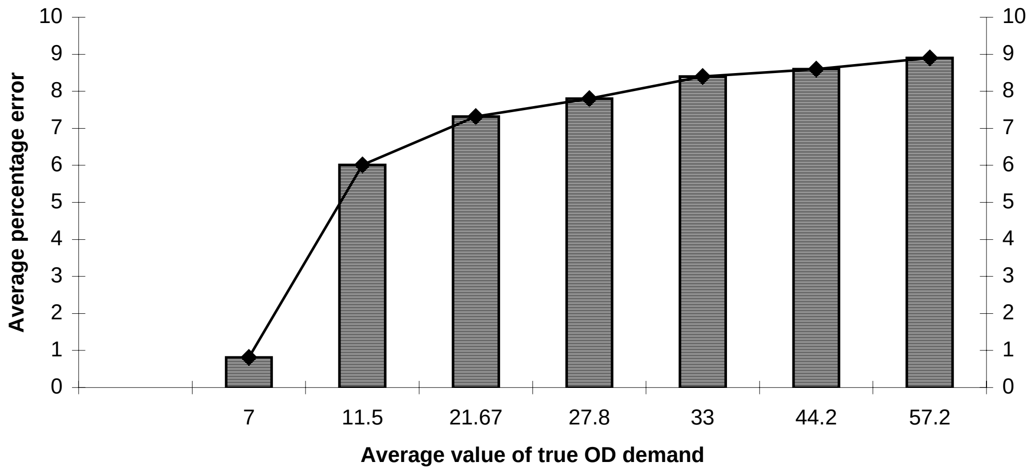


Figure 4.3 Estimation errors for OD demand values (for the cases with a maximum 20% deviations in the historical demand values from the true OD values)

A second network based on a portion of the real-world testbed network was also used. This network (Figure 4.4) is a portion of the City of Irvine between Sand Canyon and Jeffrey on 405 North and between Sand Canyon and Jeffrey on Alton. This network is cut out from the coded Paramics network of the testbed. Four zones are created and signals at all the street intersections were set up for this network. The network has 8 nodes, 8 links, 2 origins and 2 destinations. The link characteristics are shown in Table

4.3 These characteristics were used in the computation of link costs in the upper and lower problems of dynamic assignment module.

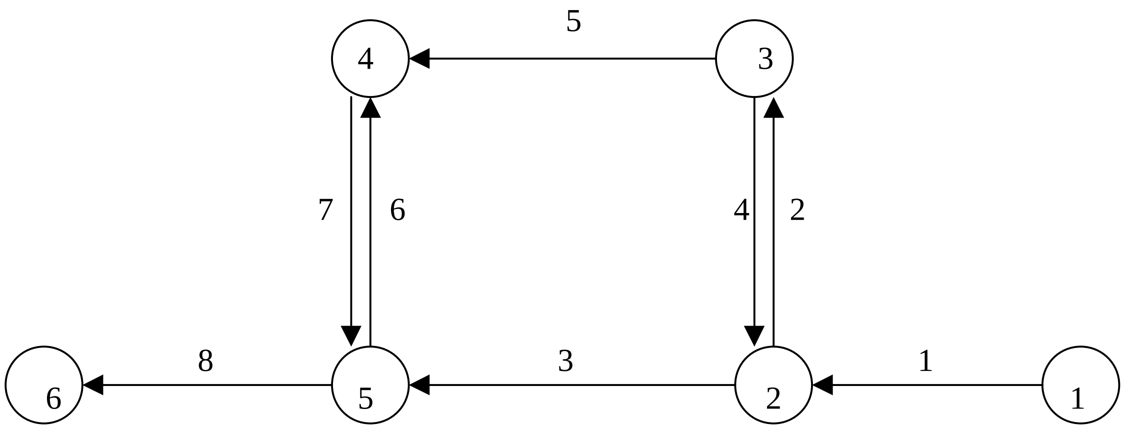


Figure 4.4 Network 2

Table 4.3 Link Characteristics (network 2)

Link	Number of Lanes	Jam Density (veh./mile/lane)	Link Length (feet)	Max Speed (feet/sec.)
1	4	160.0	1500.0	88.0
2	2	160.0	1000.0	51.33
3	4	160.0	5000.0	88.0
4	2	160.0	1000.0	51.33
5	1	160.0	4500.0	51.33
6	2	160.0	1000.0	51.33
7	2	160.0	1000.0	51.33
8	4	160.0	1500.0	88.0

The first case considered was that of 2 OD pairs. There is one origin and two destinations. OD pair 1 has node 1 as the origin and node 4 as the destination. OD pair 2 has node 1 as the origin and node 6 as the destination. The historical OD demand matrix was obtained by deviating the true OD demand values by roughly fixed percentage for each run. The percentage values were not uniform and depended on the values of the demand being integer numbers because the demand values represent the number of vehicles to be released in that time period. To avoid any kind of complexities, these values were assumed to be whole numbers to start with. Five cases (runs) are considered depending on the percentage of deviation. These cases are twenty percent, forty percent, sixty percent, eighty percent and hundred percent deviation of the true OD demand matrix. Table 4.4 (see page 54 for table layout details) shows the results of this network for various cases (historical matrix) for the dynamic OD estimation.

Table 4.4 Estimation results (Kalman Filter, network 2, case 1)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
Run 1				
10	12	20%	10.07	0.7%
15	12	20%	14.94	0.4%
12	10	16.7%	11.96	0.36%
8	10	25%	8.05	0.6%
10	12	20%	10	0%
8	10	25%	8	0%
Run 2				
10	14	40%	10.6	6.0%
15	21	40%	15.3	2.0%
12	17	41.67%	12.2	1.67%
8	11	37.5%	8.0	0%
10	14	40%	10.1	1%
8	12	50%	8.1	1.25%
Run 3				
10	16	60%	10.7	7%
15	24	60%	15.5	3.3%
12	19	58.3%	12.3	2.5%
8	13	62.5%	8.1	1.25%
10	16	60%	10.2	2%
8	13	62.5%	8.1	1.25%
Run 4				
10	18	80%	10.8	8%
15	27	80%	15.6	4%
12	22	83.3%	12.4	3.33%
8	15	87.5%	8.2	2.5%
10	18	80%	10.3	3%
8	14	75%	8.1	1.25%
Run 5				
10	20	100%	11.2	12%
15	30	100%	14.7	2.0%
12	24	100%	12.9	7.5%
8	16	100%	7.7	3.75%

10	20	100%	10.3	3%
8	16	100%	7.7	3.75%

The second case is that of two origins and one destination for the same network. This again gives two OD pairs. OD pair 1 has node 1 as the origin and node 6 as the destination. OD pair 2 has node 3 as the origin and node 6 as the destination. Table 4.5 (see page 54 for table layout details) gives the results of dynamic OD estimation for this case.

Table 4.5 Estimation results (Kalman Filter, network 2, case 2)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
Run 1				
10	12	20%	9.4	6%
15	18	20%	14.3	4.6%
12	15	25%	12.3	2.5%
18	15	16.7%	17.6	2.22%
12	15	25%	12.1	0.08%
10	12	20%	9.89	1.1%
Run 2				
10	14	40%	9.2	8%
15	21	40%	15.7	4.67%
12	17	41.67%	11.8	1.67%
18	25	38.9%	18.6	3.33%
12	17	41.67%	12.1	0.08%
10	14	40%	10.1	1%
Run 3				
10	16	60%	9.1	9%
15	24	60%	15.9	6%
12	19	58.3%	10.8	10%
18	29	61.1%	20.1	11.67%
12	19	58.3%	11.2	7.5%
10	16	60%	10.2	2%
Run 4				
10	18	80%	8.7	13%

15	27	80%	16.1	7.3%
12	21	75%	10.5	12.5%
18	33	83.3%	17.6	2.2%
12	22	83.3%	11.5	4.2%
10	18	80%	8.9	11%
<u>Run 5</u>				
10	20	100%	8.5	15%
15	30	100%	14.1	6%
12	24	100%	10.8	10%
18	36	100%	17.4	3.33%
12	24	100%	10.6	11.67%
10	20	100%	8.9	11%

The third case is that of two origins and two destinations. The network remains the same. This results in four OD pairs. OD pair 1 has node 1 as the origin and node 6 as destination. OD pair 2 has node 3 as origin and node 6 as destination. OD pair 3 has node 1 as the origin and node 4 as the destination. OD pair 4 has node 3 as the origin and node 4 as the destination. The results of the dynamic OD estimation for this case are tabulated in Table 4.6 (see page 54 for table layout details).

Table 4.6 Estimation results (Kalman Filter, network 2, case 3)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
10	12	20%	9.6	4%
15	12	20%	14.4	4%
10	12	20%	9.0	10%
8	12	50%	9.1	13.8%
12	10	16.7%	11.9	8.3%
10	12	20%	10.0	0%
8	10	25%	10.0	25%
12	10	16.7%	10.1	15.8%

8	10	25%	7.9	1.25%
10	12	20%	10.0	0%
10	8	25%	9.9	1%
12	10	16.67%	11.9	8.3%
Run 2				
10	14	40%	9.2	8%
15	21	40%	15.5	3.33%
10	14	40%	10.8	8%
8	11	37.5%	7.4	7.5%
12	17	41.67%	12.4	3.33%
10	14	40%	10.9	9%
8	11	37.5%	7.1	11.25%
12	17	41.67%	13.2	10%
8	12	50%	9.0	12.5%
10	14	40%	10.3	3%
10	14	40%	9.1	9%
12	17	41.67%	13.4	11.67%
Run 3				
10	16	60%	9.0	10%
15	24	60%	16.9	12.67%
10	16	60%	10.9	9%
8	13	62.5%	7.2	10%
12	19	58.3%	12.4	3.33%
10	16	60%	10.2	2%
8	13	62.5%	7.3	8.75%
12	19	58.3%	13.3	10.83%
8	13	62.5%	8.8	10%
10	16	60%	10.7	7%
10	16	60%	9.0	10%
12	19	58.3%	12.8	7.5%
Run 4				
10	18	80%	8.6	14%
15	27	80%	17.2	14.67%
10	18	80%	11.0	10%
8	15	87.5%	8.1	1.25%
12	22	83.3%	11.5	4.16%
10	18	80%	9.0	10%
8	15	87.5%	7.3	8.75%
12	21	75%	13.4	11.67%
8	14	75%	7.2	10%
10	18	80%	10.3	3%
10	18	80%	9.4	6%
12	22	83.3%	13.4	11.67%
Run 5				

10	20	100%	8.4	16%
15	30	100%	14.4	4%
10	20	100%	11.41	14.1%
8	16	100%	7.6	5%
12	24	100%	11.6	3.3%
10	20	100%	9.1	9%
8	16	100%	6.5	18.8%
12	24	100%	13.2	10%
8	16	100%	6.6	17.5%
10	20	100%	8.89	11.1%
10	20	100%	9.4	6%
12	24	100%	13.5	12.5%

A comparison of results obtained for various cases show that the average percentage error increases as the number of OD pairs (more specifically number of origins) increases. This can also be seen from Figure 4.5. This may be because of the fact that the more OD pairs results in a bigger OD-link incidence matrix (“B” matrix). The bigger the link incidence matrix the more unstable the mapping of it with the true OD matrix becomes. Also, it can be seen from Figure 4.3 that the average percentage errors are higher for the cases when the historical (seed) OD matrix was away from the true OD matrix by more than twenty percent. This makes it difficult for the estimated OD matrix to converge closely to the true OD matrix and hence there are higher errors in the estimated OD matrix.

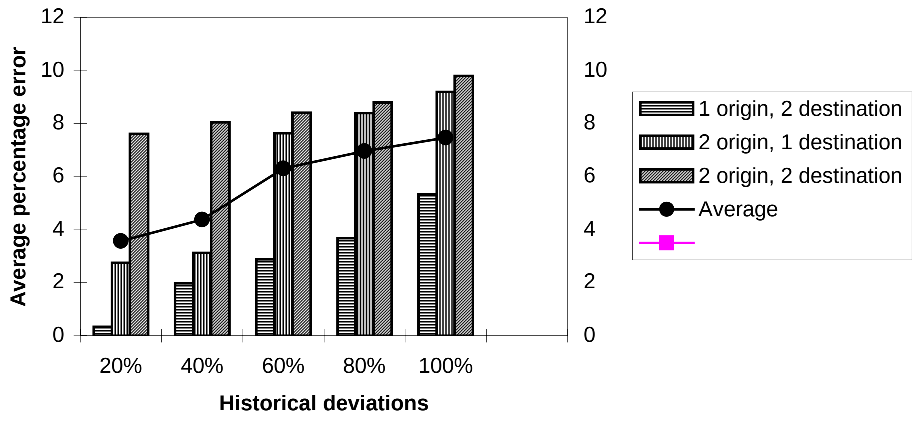


Figure 4.5 Comparison of various cases using Kalman Filter

4.6 RESULTS USING LEAST SQUARES FOR DYNAMIC ODE

The network used was composed of six links and four nodes. The network considered has two OD pairs. The link characteristics are shown in Table 4.1. The network is shown in Figure 4.2. These characteristics were used in the computation of the link cost in the upper and lower problems of the dynamic traffic assignment module. OD pair 1 has node 1 as the origin and node 4 as the destination. OD pair 2 has node 1 as the origin and node 3 as the destination. Using the Dynamic Traffic Assignment module, the dynamic OD-link incidence matrix (“B” matrix) and the ideal link densities are obtained. Three different time steps are used. The results for dynamic OD demand estimation are tabulated in Table 4.7 (see page 54 for table layout details).

Table 4.7 Estimation Results (Least Squares, network 1)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
5	5	0%	5	0%
10	9	10%	10	0%
5	4	20%	5	0%
7	6	14.3%	6.9	1.43%
9	8	11.1%	9.0	0%
6	5	16.7%	6.0	0%
<u>Run 2</u>				
13	12	7.7%	12.73	2.1%
12	11	8.5%	12.0	0%
11	10	9.09%	10.6	3.6%
10	9	10%	10.0	0%
11	10	9.09%	10.62	3.5%
12	11	8.5%	11.78	1.8%
<u>Run 3</u>				
20	23	15%	20.8	4.0%
30	32	6.67%	30.4	1.33%
20	22	10%	21.1	5.5%
25	22	12%	24.3	2.8%
15	18	20%	15.9	6.0%
20	18	10%	19.1	4.5%
<u>Run 4</u>				
30	35	16.67%	31.8	6.0%
35	40	14.2%	36.4	4.0%
30	35	16.67%	31.2	4.0%
35	40	14.2%	36.6	4.6%
38	45	18.4%	40.0	5.3%
30	36	20%	31.4	4.67%
<u>Run 5</u>				
40	45	12.5%	41.6	4.0%
45	50	11.1%	47.2	4.9%
42	48	14.2%	43.8	4.3%
44	50	13.7%	46.6	5.9%
48	55	14.6%	50.3	4.8%
46	52	13.1%	48.9	6.3%
<u>Run 6</u>				
50	58	16%	52.8	5.6%
58	65	12.1%	61.2	5.5%
60	65	8.3%	62.7	4.5%

55	62	12.7%	57.9	5.3%
62	65	4.8%	65.0	4.8%
58	65	12.1%	61.7	6.3%

The results show that the Least Squares estimated the OD demand values very close to the true OD demand values.

The second network used was the portion of the Irvine network with previous section. The network is shown in Figure 4.4. The link characteristics are the same as shown in Table 4.3. These characteristics were used in the computation of link costs in the upper and lower problems of dynamic assignment module.

The first case considered was that of 2 OD pairs. There is one origin and two destinations. OD pair 1 has node 1 as the origin and node 4 as the destination. OD pair 2 has node 1 as the origin and node 6 as the destination. Table 4.8 (see page 54 for table layout details) shows the results of this case for dynamic origin/destination demand estimation.

Table 4.8 Estimation results (Least Squares, network 2, case 1)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
10	12	20%	10	0%
15	12	20%	15.2	1.33%
12	10	16.7%	12.0	0%

8	10	25%	8	0%
10	12	20%	10	0%
8	10	25%	8	0%
Run 2				
10	14	40%	10.0	0%
15	21	40%	15.4	2.67%
12	17	41.67%	11.8	1.4%
8	11	37.5%	8.1	1.25%
10	14	40%	10.0	0%
8	12	50%	8.0	0%

The second case is that of two origins and one destination. The network is the same as Figure 4.4. This again gives two OD pairs. OD pair 1 has node 1 as the origin and node 6 as the destination. OD pair 2 has node 3 as the origin and node 6 as the destination. Table 4.9 (see page 54 for table layout details) gives the results of dynamic OD estimation for this case.

Table 4.9 Estimation results (Least Squares, network 2, case 2)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
Run 1				
10	12	20%	10	0%
15	18	20%	14.6	2.67%
12	15	25%	12	0%
18	15	16.7%	17.4	3.33%
12	15	25%	12	0%
10	12	20%	9.8	2%
Run 2				
10	14	40%	10.1	1.0%
15	21	40%	14.7	2.0%
12	17	41.67%	12.0	0%
18	25	38.9%	17.4	3.33%
12	17	41.67%	12.0	0%

10	14	40%	10.1	1%

The third case is that of two origins and two destinations for the same network (Figure 4.4). This results in four OD pairs. OD pair 1 has node 1 as the origin and node 6 as destination. OD pair 2 has node 3 as origin and node 6 as destination. OD pair 3 has node 1 as the origin and node 4 as the destination. OD pair 4 has node 3 as the origin and node 4 as the destination. The results of the dynamic OD estimation for this case are tabulated in Table 4.10 (see page 54 for table layout details).

Table 4.10 Estimation results (Least Squares, network 2, case 3)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
10	12	20%	9.6	-4.0%
15	12	20%	14.5	-3.33%
10	12	20%	9.2	-8.0%
8	12	50%	8.2	2.5%
12	10	16.7%	11.9	-0.833%
10	12	20%	10.0	0%
8	10	25%	9.2	15.0%
12	10	16.7%	10.6	-11.66%
8	10	25%	8.2	2.5%
10	12	20%	10.0	0%
10	8	25%	9.9	-1.0%
12	10	16.67%	12.0	0%
<u>Run 2</u>				
10	14	40%	9.4	-6.0%
15	21	40%	15.5	3.33%
10	14	40%	10.3	3.0%
8	11	37.5%	7.5	-6.25%
12	17	41.67%	12.4	3.33%
10	14	40%	10.0	0%
8	11	37.5%	7.4	-7.5%
12	17	41.67%	12.3	2.5%

8	12	50%	8.3	3.75%
10	14	40%	10.3	3.0%
10	14	40%	9.2	-8.0%
12	17	41.67%	13.1	9.16%

A comparison of results obtained for various cases show that the average percentage error increases as the number of OD pairs (more specifically number of origins) increases. This is again because of the fact that the more OD pairs results in a bigger OD-link incidence matrix ("B" matrix). The bigger the link incidence matrix the more unstable the mapping of it with the true OD matrix becomes. Also, it can be seen that the average percentage errors are higher for the cases when the historical (seed) OD matrix was away from the true OD matrix by more than twenty percent. This makes it difficult for the estimated OD matrix to converge closely to the true OD matrix and hence there are higher errors in the estimated OD matrix.

4.7 COMPARISON OF RESULTS OBTAINED FROM KALMAN FILTER AND LEAST SQUARES

From a comparison of the above results, it can be seen that the Least Squares method performs better than the Kalman Filter method for dynamic origin/destination demand estimation. Figure 4.6 shows a comparison of the average percentage errors for the estimated OD demand matrix obtained for Kalman Filter and the Least Squares methods. These results are for all the cases where the historical (seed) OD demand matrix was deviated not more than twenty percent from the true origin/destination demand matrix. It can be seen that the average percentage errors for the estimated OD matrix using the Least Squares method is much smaller compared to the Kalman Filter method for the same case. This may be due to the fact that the Least Squares method is a one-step method which tries to minimize the errors while Kalman Filter tries to minimize the errors in an indirect fashion. Also Kalman Filter is a minimum mean square estimator of the state variables assuming auto-correlation, while least squares does not assume auto-correlation. The auto-correlation assumed for the Kalman Filter method was 0.001.

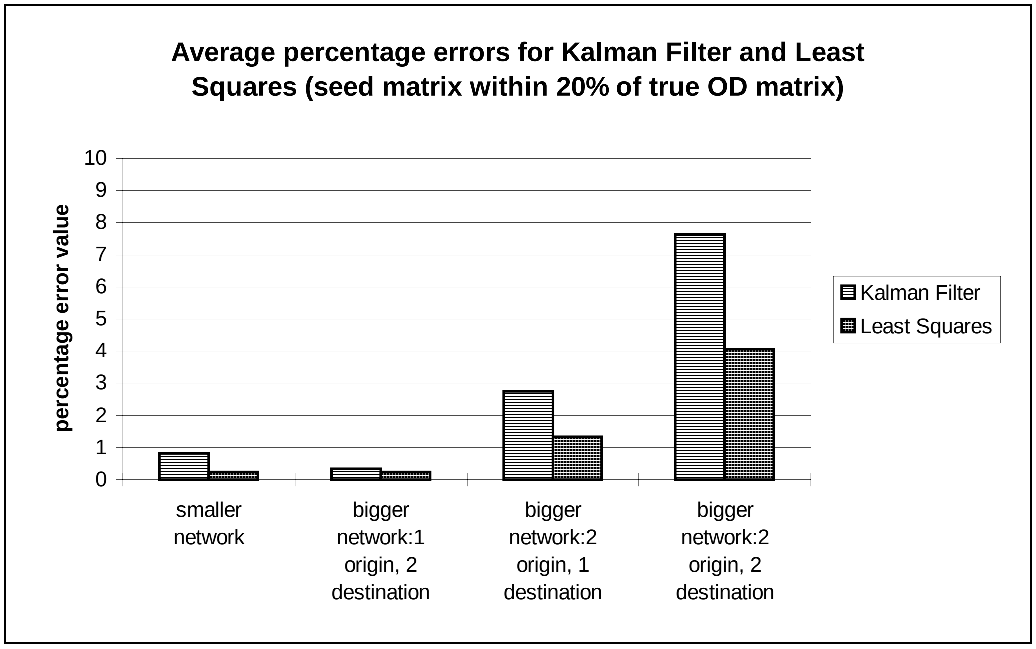


Figure 4.6 Comparison results obtained from Kalman Filter and Least Squares methods

Similarly, for the cases where the historical (seed) OD demand matrix was deviated more than twenty percent, the average percentage errors for the Least Squares method are much smaller than the average percentage errors from Kalman Filter method (Figure 4.7). This is again due to the fact that the Least Squares method is a more direct method which tries to minimize errors in one-step while Kalman Filter minimizes errors in an indirect fashion.

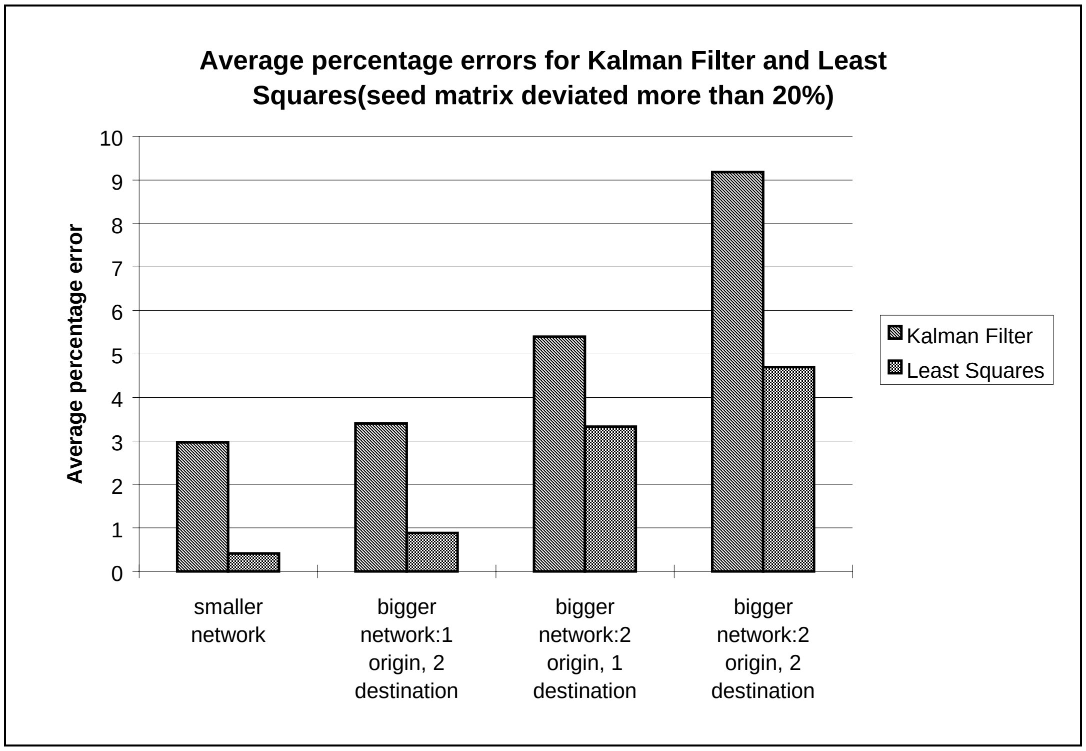


Figure 4.7 Comparison of Kalman Filter and Least Squares results

CHAPTER 5. DYNAMIC OD ESTIMATION USING DYNASMART

5.1 INTRODUCTION TO DYNASMART

This chapter provides a brief synopsis of both the practical implementation and the theoretical background of the components of DYNASMART. Each synopsis consists of major subroutines, followed by the mathematical relationships underlying the theoretical basis for that subroutine. It is beyond the scope of this report to document the microscopic details of the operations of DYNASMART. For more information on the operations of DYNASMART, please refer to Jayakrishnan (1991).

DYNASMART is appropriate for our research because of the following functional capabilities:

1. It has the ability to model the route choice of drivers with and without access to ATIS information.
2. responsiveness to dynamic O-D information available to the controllers as reported by the ATIS and other sources.
3. ability to track location of drivers, both those who receive advice from control center and those who do not.
4. ability to predict time-dependent impedance (travel time) based on the assignment results, and provide feedback to the control center that may be used in the assignment of vehicles.

5. ability to model real-time traffic control strategies in the network, such as signal settings and ramp metering, that would directly affect the network delays and route dynamics;
6. ability to comprehensively model both freeway traffic and surface street traffic.

The DYNASMART (DYnamic Network Assignment-Simulation Model for Advanced Road Telematics) framework is a mesoscopic simulation-based approach to the problem of modeling traffic flow under information. Mesoscopic simulations are simulations which fall between microscopic and macroscopic simulations. DYNASMART combines, in a modular structure, the three necessary elements of dynamic traffic simulation under information: (1) traffic flow simulation, (2) driver behavior modeling, and (3) dynamic network path processing.

There is also a fourth component - a main module which controls the overall flow of the simulation, handles the external interfaces, and updates the simulation clock. A graphical representation of the modular structure and operation of DYNASMART is shown in Figure 5.1.

DYNASMART is written in the FORTRAN programming language because this language is adept at performing extremely fast internal computations. The wide use of FORTRAN in engineering also facilitates the continual updating and improvement of the program in future research projects.

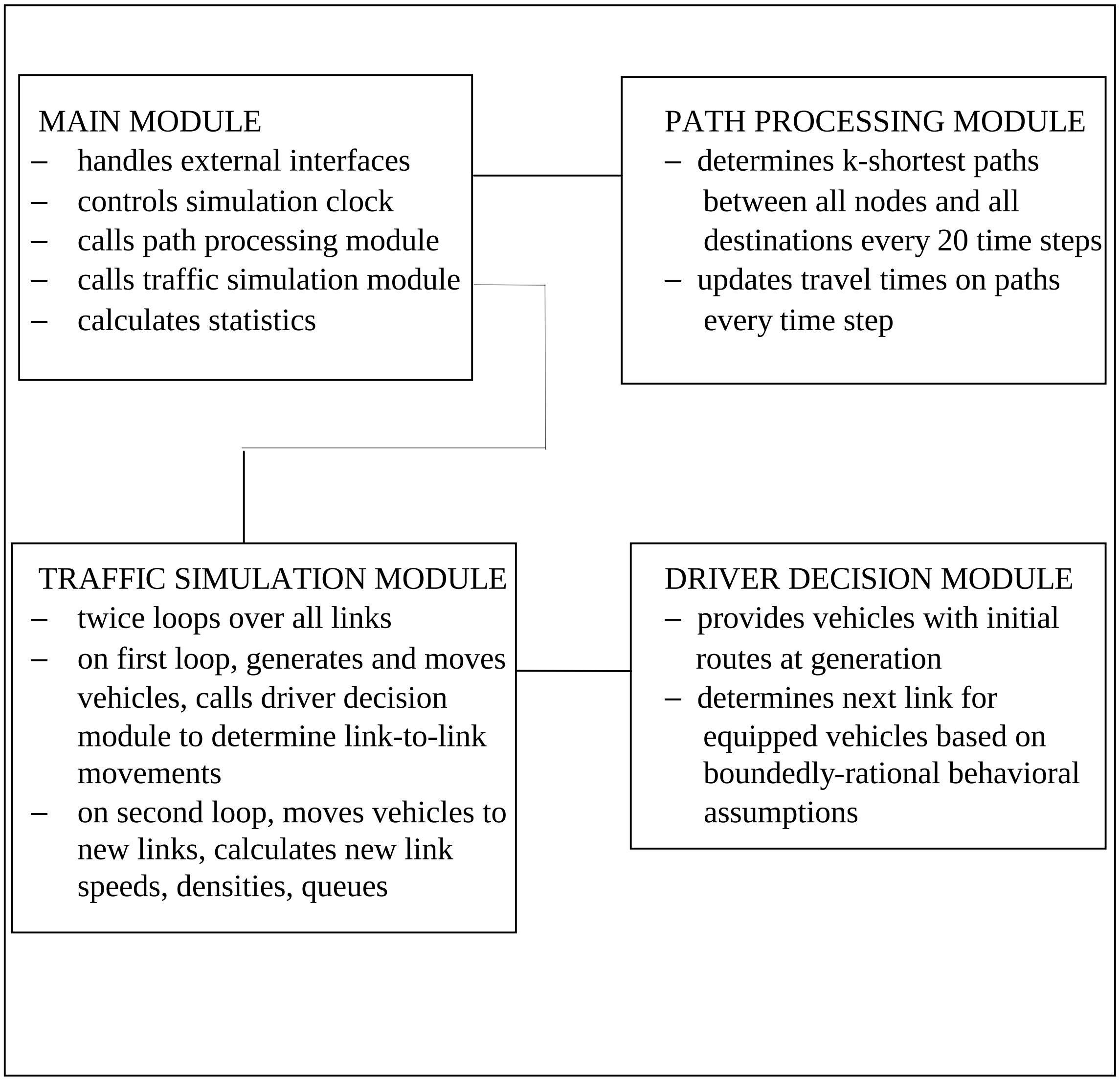

MAIN MODULE
– handles external interfaces
– controls simulation clock
– calls path processing module
– calls traffic simulation module
– calculates statistics
PATH PROCESSING MODULE
– determines k-shortest paths between all nodes and all destinations every 20 time steps
– updates travel times on paths every time step
TRAFFIC SIMULATION MODULE
– twice loops over all links
– on first loop, generates and moves vehicles, calls driver decision module to determine link-to-link movements
– on second loop, moves vehicles to new links, calculates new link speeds, densities, queues
DRIVER DECISION MODULE
– provides vehicles with initial routes at generation
– determines next link for equipped vehicles based on boundedly-rational behavioral assumptions

Figure 5.1 Modular structure of DYNASMART

This chapter consists of five sections in addition to this introduction. Section 5.2 will discuss the operation of the main module. Section 5.3 through 5.5 will discuss the operation of the traffic flow module, the driver behavior module, and the path processing module, respectively.

5.2 MAIN MODULE

The main module, hereafter referred to as MAIN, is responsible for data input, initialization the simulation, calling the other modules at the appropriate times, updating the simulation clock, and terminating the simulation when the clock has reached the simulation time limit. After the simulation is terminated, MAIN computes the necessary statistics and creates the output files.

The required input data consists of network data, and simulation parameters. Network data includes node connectivity, link characteristics, zonal information, and simulation time slice limits. Demand data consists of inter-zonal vehicle flows for each discrete time slice. Simulation parameters include the fraction of drivers equipped with information, the mean value for the indifference bandwidth, the time span over which statistics are collected, and the frequency for updating the k-shortest paths.

The simulation is clock-driven as opposed to event-driven. Thus, with each pass through the MAIN controlling loop, the simulation clock is updated by a constant time interval equal to one-tenth of a minute (six seconds) of simulation time. This time-interval is modifiable.

For example, during a ninety-minute simulation, travel statistics will be collected only for vehicles which enter the network between the fifteenth and the sixtieth minute. This allows fifteen minutes at the start of the simulation to “warm up” the network (i.e., allow links to reach reasonable densities), and one-half hour at the end of the simulation for all tagged vehicles to exit the network. Of course, the simulation time as well as the period of “vehicle tagging” can be set to different values by the user.

5.3 TRAFFIC FLOW MODULE

The traffic flow module is responsible for the generation and movement of individual vehicles through the network, and the updating of link traffic conditions. This component consists of three subroutines, hereafter referred to as PARTCO, CHANGE, and RESTORE. PARTCO is responsible for generating vehicles, moving the vehicles to their updated positions on the network, and updating link speeds, densities, and queues. CHANGE and RESTORE are responsible for updating network characteristics based on incidents.

PARTCO performs five specific functions:

1. vehicle generation and entry onto the network;
2. movement of vehicles within links;
3. calculation of link-to-link vehicle flux constraints;
4. movement of vehicles from link to link, and

5. updating of link conditions at the end of the time step.

Unlike other network models, DYNASMART does not generate vehicles at centroids. Instead, vehicles are generated directly onto network links which are specified in the input data as containing traffic generation based on either the upstream or downstream node. These nodes, in turn are specified as existing in one of the zones for which inter-zonal demand has been provided. As each individual vehicle is generated and loaded onto the network, it is provided with an initial path from the BEGINRT module (section 5.4). Vehicles are also identified as equipped or unequipped, and as tagged or untagged.

Unequipped vehicles (no ATIS information) remain on the path provided by BEGINRT for their entire journey. Equipped vehicles can switch paths enroute based on the binary decision rule discussed in Section 5.4. Tagged vehicles are utilized to calculate aggregate system-performance statistics. Untagged vehicles behave in the same manner as tagged vehicles, but their statistics are not collected.

Vehicles which are already on the network are moved by PARTCO at the prevailing link speeds. When a vehicle reach the end of a link, a decision is required as to which downstream link the vehicle should enter. GETLINK is thus called; as explained in Section 5.4, GETLINK considers boundedly-rational driver behavior to communicate to PARTCO the next link onto which to move a vehicle. After PARTCO has determined the number of vehicles to be moved between links, it must calculate maximum link flux rate to avoid an unrealistically high volume of link-to-link vehicle movements. this could happen in the event of quick dissipation of downstream traffic, such as occurs following the conclusion of an incident.

In the case where vehicles from several sources are ready to enter the same link, PARTCO must determine the entrance priority for all the relevant vehicles according to a specified protocol. For example, vehicles entering a link as a result of demand generation are given priority over vehicles entering a link from an upstream link. PARTCO loops over the entire set of links in the network twice. In the first loop, at each link, the module generates and loads the new vehicles, moves the existing vehicles along the link, determines which vehicles have reached the end of the link, and calls the driver decision component to determine link-to-link demands. In the second loop, PARTCO determines the entrance priority of the vehicles onto their new links, moves the vehicle onto the new links, and recalculates the link densities, speed, and queue lengths, based on the new vehicle positions, which will be applied in the next time step.

To calculate link flow characteristics based on vehicle positions, PARTCO utilizes the equations of macroscopic traffic flow. These equations are derived from the basic conservation equation, which is a partial differential equation relating the rate of change of traffic flow over space to the rate of change in traffic density over time. This equation is shown below:

$$\frac{\partial q}{\partial x} + \frac{\partial k}{\partial t} = g(x, t)$$

where, q = traffic flow in vehicles per unit time period

k = traffic density in vehicles per unit distance

g = net generation of vehicles per unit time and distance

x = distance

t = time

Since DYNASMART keeps track of the positions of individual vehicles, PARTCO is able to utilize to finite difference form of the conservation equation to update link densities in each time step. The finite difference form of the equation is expressed as:

$$K_j^{t+1} = K_j^t + \frac{\Delta t}{\Delta X_j}\left[q_{ij}^t - q_{oj}^t + I_j^t - O_j^t\right]$$

where, K_j^t = vehicle density of link during the t^{th} time step

Δt = simulation time increment

Δx_j = link length

q_{ij}^t = total inflow traffic volume from upstream links

q_{oj}^t = total outflow traffic volume to downstream links

I_j^t = volume of external traffic entry

O_j^t = volume of traffic exiting

Thus, the vehicle density of a link in each time step is a function of the density in the previous time step and the link-to-link movements of vehicles during the time step, as well as the addition of vehicles through generation.

Once PARTCO has updated link densities, it needs to update the link speed to calculate vehicle positions in the next time step. The equation to calculate speed as a function of link density is a modified form of Greenshield's equation, or:

$$v_j^t = \left(v_{fj} - v_o\right)\left(1 - \frac{k_j^t}{k_o}\right)^{\alpha} + v_o$$

where,	v_j^t = mean speed prevailing in segment j at time t
	v_{fj} = mean free-flow speed in segment j
	v_o = minimum speed at jam density
	k_j^t = prevailing vehicular density
	k_o = jam density
	α = a parameter alpha

This equation provides for a non-linear relationship between density and speed, where the value of alpha determines the extent of the non-linearity of the function. The original form of Greenshield's equation is the special case of the above equation where v_o is zero and α is one.

By providing for a non-zero minimum speed of v_o at jam density, this modified version of Greenshield's equation eliminates the problem inherent to the original form of the equation. This problem occurs when the link speed is zero at jam density, thus effectively "shuts down" the simulation by not permitting vehicles to move in the following time step.

5.4 DRIVER BEHAVIOR MODULE

This module performs a simple but realistic and flexible emulation of boundedly-rational driver behavior. As IVNS-equipped vehicles approach a node, a decision needs to be made as to which downstream link the vehicle will enter; this in turn is a function of the path to which the vehicle is assigned. This section will describe how DYNASMART emulated the driver-decision process. The driver behavior module consists of two subroutines, hereafter referred to as BEGINRT and GETLINK.

BEGINRT is called as any vehicle, equipped or unequipped, is generated and loaded onto a network link. This subroutine chooses the initial route to which a vehicle is assigned. This is the shortest path which exists between that link and the vehicle's destination at the end of the simulation warm-up period. Unequipped vehicles remain on this path for the entire simulation; equipped vehicles can switch paths via the GETLINK subroutine.

GETLINK is called as an equipped vehicle approaches the end of a link; its function is to return the next link onto which the vehicle should move. This subroutine first calculates the travel time to the destination on the vehicle's currently assigned path; then it compares this travel time with the travel time to the destination from that node on the minimum of the k-shortest paths (as described in section 5.5). If the driver behavior rules warrant, a change in paths is performed. A path change is dictated by the binary switch:

$$\delta_j(k) = \begin{cases} 1, & \text{if } \left[TTC_j(k) - TTB_j(k)\right] > \text{ max.}\left(\mu_j \bullet TTC_j,\ \tau_j\right) \\ 0, & \text{otherwise} \end{cases}$$

where, for driver j:

$\delta_j(k)$ = 1 indicates a route switch, 0 indicates no route switch

$TTC_j(k)$ = current trip time from node k to destination

$TTB_j(k)$ = trip time from node k to best alternate path

μ_j = relative indifference band threshold

τ_j = minimum improvement needed for switch

The value of μ_j is specified for each vehicle at the time of that vehicle's generation; the value is chosen from a triangular distribution with a user-specified mean. The specific case where both μ and τ are equal to zero is known as "myopic" switching, where the driver switches routes if there is any shorter route. Myopic switching can be considered as the modeling of a prescriptive system, in which drivers are compelled to follow the prescribed route.

If the rules indicate a switch, the vehicle is reassigned to the new shortest path, and GETLINK returns to PARTCO the next link on the new path. Otherwise, the vehicle remains on the previous path, and GETLINK returns to PARTCO the next link on the previous path. It is important to note that all routing decisions in GETLINK are performed only with information concerning the current link travel times which prevail at the time the decision is being made. There is no effort to predict and consider the future travel times which will prevail on each link of the path when the vehicle actually reaches that link. In the periods of high congestion, the variance between current and future travel times could be quite extreme, with significant implications for system effectiveness and reliability. This in turn could have serious negative consequences for driver confidence and public acceptance of ATIS.

This behavior modeling framework is also limited by considering total path travel time as the only determining factor in the route-switching decision. Other factors which may be very important in route choice, such as the number of signalized intersections on each path, are not considered. However, the ability of DYNASMART to determine k-shortest paths facilitates the addition of such factors to the decision-rule component in the future.

5.5 PATH PROCESSING MODULE

This module is the most complex component of DYNASMART. The path processing module is responsible for enumerating the k-shortest paths between every node and every destination, calculating and updating the total travel times on these paths in every time step, and re-enumerating the paths at regular intervals based on the updated link travel times caused by congestion. The path-processing module consists of three subroutines, hereafter referred to as ADDCHAIN, KSHORT, and ROUTETM.

The first subroutine called by MAIN in each time step in ADDCHAIN. ADDCHAIN sums the current trip times on segment arcs. A segment arc is a continuous series of links over which no route change is permissible, and thus segment arcs can be treated as a single unit in the calculation of route travel times. MAIN then calls either KSHORT or ROUTETM, depending on the number of the time intervals. KSHORT loops over the destinations and builds a tree of k-shortest paths from every destination to every node. However, KSHORT is not called at every time step as this would consume substantial amounts of computer time. In the simulations documented in this research, KSHORT is called every one minute of simulation-clock time. and the number of different paths enumerated between each destination and each node at each call of KSHORT is two.

KSHORT employs an efficient and sophisticated label-correcting algorithm to find the k-shortest paths based on the updated link and arc travel times. The details on the structure of the label-correcting algorithm are presented by Jayakrishnan (1991). For the time intervals in which KSHORT is not called, the subroutine ROUTETM is called, which loops over the destinations and updates the travel times on the existing k-shortest paths. It is not expected that these paths will still necessarily be the actual k-shortest paths, but this approximation is preferable to re-enumerating the paths at every time step.

The k-shortest paths enumerated by the algorithm are often very similar to each other, with only a few links differentiating all of the paths. This severely limits the route choices available to the driver. It may be desirable to eventually modify the path-finding algorithm to provide k-shortest routes which all differ from each other by a specified parameter (such as ten percent of the links).

5.6 CONCEPTUAL FRAMEWORK FOR ODE USING KALMAN FILTER

This formulation of the dynamic origin/destination demand estimation uses the Kalman Filter method. The Kalman Filter estimation procedure is used for determining an optimal estimate of origin/destination demands based on a set of noisy but unknown measurements of the section density. The section densities and the origin/destination demands are assumed to be linearly related. Following if the framework of this problem formulation.

The state equation of the model in matrix notation is

$$q(k+1) = \sum_{h \in N_h} F(h)q(h) + w(k)$$

$$E[w(k)] = 0$$

$$E[w(i)w(k)] = \begin{cases} 0, & i \neq k \\ Q(k), & i = k \end{cases}$$

where,

$$k = timesteps \ (1,2,3,....T)$$

The state variable $q(k+1)$ denotes the vector of departure OD demands for every OD pair at origin node O at time $k+1$. The index h is a temporal index that represents every time step that is relevant to the current time step. The matrix $F(h)$ contains all the regression coefficients which relate the current OD demands to previous OD demands and also expresses the relationships across OD pairs. The error vector $w(k)$ is assumed to be white noise with zero mean, and $Q(k)$ is a $\#(N_{ij}) \times \#(N_{ij})$ symmetric and positive semi-definite covariance matrix.

The measurement equation is

$$y(k+1) = \sum_{h \in N_n} B(k+1)q(k+1) + v(k+1)$$

$$E[v(k)] = 0$$

$$E[v(i)v(k)] = \begin{cases} 0, & i \neq k \\ R(k), & i = k \end{cases}$$

$$E[v(i)w(k)] = 0$$

The equation relates vector $y(k)$, a $l \times T$ vector composed of the section densities of all the links in the network, with the origin/destination demands. Here, $v(k + 1)$ is white noise with zero mean and known covariance $R(k)$ and is uncorrelated with $w(k)$. The matrix $B(k + 1)$ maps the state variables to the measurements.

One of the difficult values to obtain from the measurement equation is the OD/link incidence matrix, denoted by $B(h)$. Most of the dynamic OD matrix estimation schemes which require this matrix have always assumed that this variable was available, and the issue of deriving such a variable has always been circumvented. This difficulty is overcome by the use of DYNASMART or DTA as discussed in the last chapter.

Using the DYNASMART module, the dynamic OD-link incidence matrix and the link densities are obtained. These are then used to arrive at the optimal estimate of the dynamic OD demand values. We use DYNASMART to both simulate real world and derive OD-link incidence using historical seed matrix.

We use the 1-Step Kalman Filter for off-line testing which can be later modified to be rolling horizon. One network that was used was composed of twenty-five links and twenty-one nodes. This is a real network cut out of the Paramics Irvine network. The network consists of a section of 405 North Freeway, Jeffrey and Sand Canyon. The network has four zones. Two zones are origin zones and the other two are destination zones. Origin zones have one link each as the demand generation link. The network is shown in Figure 5.2. The link characteristics are shown in Table 5.1. This is the DYNASMART network corresponding to the DTA network (Figure 4.4). There are four OD pairs. OD pair 1 has zone 1 as the origin and zone 3 as the destination. OD pair 2 has zone 1 as the origin and zone 4 as the destination. OD pair 3 has zone 2 as the origin zone and zone 3 as the destination zone. OD pair 4 has zone 2 as the origin zone and zone 4 as the destination zone. The demand are loaded onto the network in four time intervals of 5 minutes each.

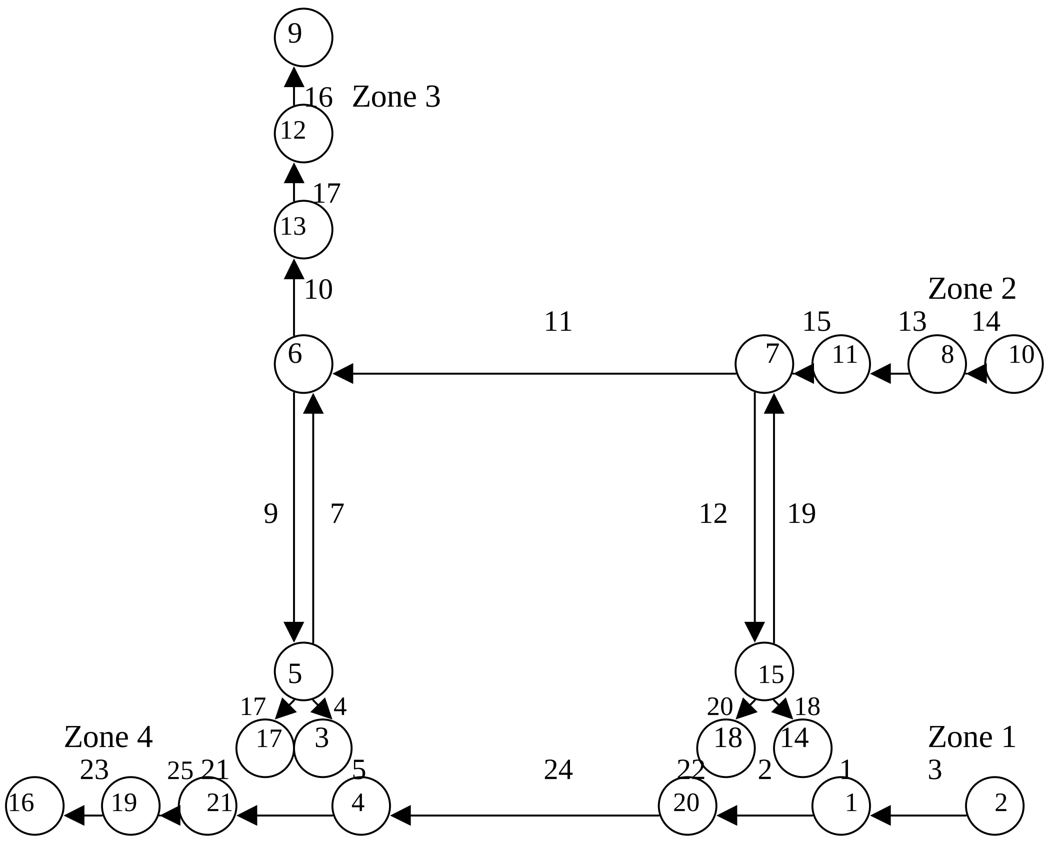


Figure 5.2 Network 3

Table 5.1 Link Characteristics (network 3)

Link	Number of Lanes	Jam Density (per lane)	Link Length (feet)	Max Speed (feet/sec)
1	2	160.0	1474	44.0
2	6	160.0	1580	95.33
3	6	160.0	2287	95.33
4	3	160.0	269	44.0
5	6	160.0	1161	44.0
6	6	160.0	1340	95.33
7	3	160.0	2054	66.0
8	6	160.0	1382	44.0

9	4	160.0	2054	66.0
10	3	160.0	842	66.0
11	3	160.0	5379	66.0
12	4	160.0	911	66.0
13	4	160.0	1279	66.0
14	3	160.0	543	66.0
15	1	160.0	263	66.0
16	5	160.0	377	66.0
17	4	160.0	572	66.0
18	3	160.0	185	44.0
19	3	160.0	911	66.0
20	6	160.0	1198	44.0
21	6	160.0	635	44.0
22	6	160.0	1003	44.0
23	6	160.0	3000	95.33
24	6	160.0	4166	95.33
25	6	160.0	4275	95.33

Table 5.2 has four columns: true OD demand matrix, an arbitrary historical OD demand matrix used to initialize the dynamic OD estimation algorithm, the resulting estimated OD demand matrix, and the percent error from the true OD demand matrix. Sixteen rows for each run corresponds to the four OD pairs times the four OD demand intervals. Row 1 is demand for OD pair 1 for time interval 1. Row 2 is demand for OD pair 2 for time interval 1. Row 3 is demand for OD pair 3 for time interval 1. Row 4 is OD pair 4 for time interval 1. Row 5 is OD pair 1 for time interval 2. Row 6 is OD pair 2 for time interval 2. Row 7 is demand for OD pair 3 for time interval 2. Row 8 is OD pair 4 for time interval 2. Row 9 is demand for OD pair 1 for time interval 3. Row 10 is OD pair 2 for time interval 3 and so on.

The first case considered is that of a low OD demand loading onto the network. Low loading does not cause any congestion (any amount of traffic delays) onto the network. The results are summarized in Table 5.2 (see page 54 for further table layout details) below.

Table 5.2 Estimation results (Kalman Filter, real network, low loading)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
Run 1				
60	50	16.67%	66.5	10.8%
80	70	12.5%	77.4	3.25%
80	70	12.5%	75.2	6.0%
50	60	20%	57.1	14.2%
60	50	16.67%	57.8	3.6%
70	60	14.3%	72.1	3%
80	70	12.5%	80	0%
50	60	20%	43.7	12.6%
60	50	16.67%	60.3	0.5%
50	60	20%	60.2	20.4%
60	60	0%	60	0%
80	80	0%	80	0%
40	50	25%	44	10%
60	80	33.3%	60	0%
70	80	14.3%	80.1	14.4%
75	75	0%	75	0%
Run 2				
60	50	16.67%	64.1	6.8%
80	60	25%	81.5	1.9%
80	60	25%	71.2	11.0%
50	80	60%	60.1	20.2%
60	40	33.3%	57.1	4.8%
70	50	28.6%	62.1	11.3%
80	50	37.5%	84	5.0%
50	70	40%	41.6	16.8%
60	80	33.3%	60.3	0.5%

50	80	60%	50.1	0.2%
60	80	33.3%	60.5	0.83%
80	50	37.5%	81.8	2.3%
40	60	50%	40.1	2.5%
60	80	33.3%	71.3	18.8%
70	50	28.6%	65.1	7%
75	50	33.3%	70.7	5.7%

The results indicate that the dynamic OD estimation was able to track the OD demand values to within 94.5 percentage accuracy when the historical OD demand values are deviated from the true OD demand values by an average of 14 percent. Also, when the average historical OD demand values are deviated by an order of 36 percent, the average OD demand values are estimated within 92 percentage accuracy.

The second case considered is that of a heavy loading of OD demand values on the network. Heavy loading leads to congestion and results in average traffic delays of 0.8 minutes (or 48 seconds) for a vehicle to traverse the network. The results for this case are summarized in Table 5.3 (see page 54 for table layout details) below.

Table 5.3 Estimation results (Kalman Filter, real network, heavy loading)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
155	135	12.9%	131.3	15.3%
188	188	0%	206.5	9.8%
188	148	21.3%	208.1	10.7%

167	147	11.9%	172.2	3.1%
256	256	0%	248.4	2.9%
137	137	0%	128.1	6.5%
147	157	6.8%	131.4	10.6%
27	27	0%	32.1	18.9%
136	116	14.7%	116	14.7%
129	139	7.8%	139	7.8%
238	218	8.4%	238	0%
119	109	8.4%	109	8.4%
156	156	0%	156	0%
119	119	0%	119	0%
138	138	0%	138	0%
129	129	0%	129	0%
Run 2				
155	205	35.5%	157	1.3%
188	220	17.1%	172	2.1%
188	150	20.2%	226	20.2%
167	200	19.8%	141	15.6%
256	200	28%	231	9.8%
137	160	16.8%	154	12.4%
147	180	22.4%	108	26.5%
27	75	177.7%	22	18.5%
136	150	10.3%	116	14.7%
129	150	16.3%	139	7.8%
238	200	16%	218	8.4%
119	150	26.1%	109	8.4%
156	120	23.1%	156	0%
119	150	26.1%	119	0%
138	150	8.7%	135	2.2%
129	150	16.3%	129	0%

The results show that the dynamic OD demand estimation was able to track the OD demand to within 93.3 percentage accuracy when the historical demand values are away from the true OD demand values by more than 5 percent. The average estimation accuracy drops to 90 percentage when the historical OD demand values are deviated from the true OD demand values by more than 30 percent.

A comparison of the two cases show that the average percentage error of estimation of dynamic OD demand values are more for the heavy loading of the network case (Figure 5.3). This is in conjunction with the fact that when a heavy demand is loaded onto the network, it results in traffic congestion which results in more than jam link densities and this makes it difficult in back-tracking of true OD demand value.

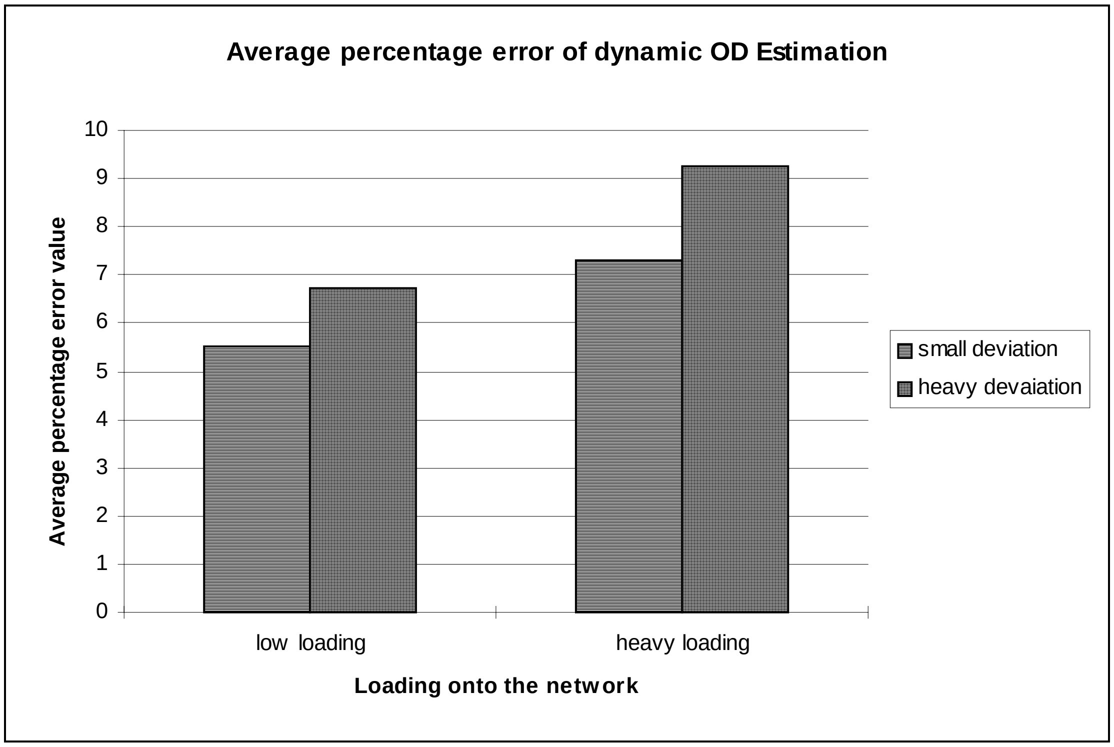


Figure 5.3 Comparison of low and heavy loading on the network (Kalman Filter, real network)

Another network was chosen. This network is smaller than the real network. The network has seven nodes, nine links and three zones. There is one origin zones and two destination zones. This results in two OD pairs. OD pair 1 has zone 1 (node 1) as the origin zone and zone 3 (node 3) as the destination zone. OD pair 2 has zone 1 (node 1) as the origin zone and zone 2 (node 4) as the destination zone. A graphical representation of the network is shown in Figure 5.4. The link characteristics are shown in Table 5.4. This network is similar to the network 1 (Table 4.1 and Figure 4.2) except that this network has three more links. These three links are added due to DYNASMART requirement that origins and destinations zones should have a minimum of one link each. As a result three more links were added onto the network 1 to make it DYNASMART compatible.

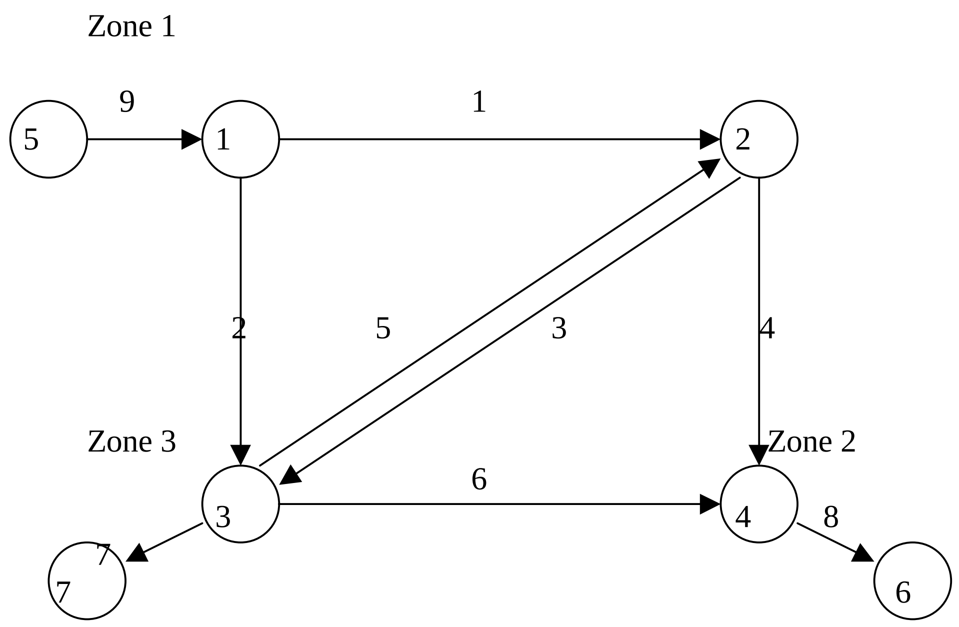


Figure 5.4 Network 4

Table 5.4 Link Characteristics (network 4)

Link	Jam Density (veh/mile)	Link Length (feet)	Max. Speed (feet/sec.)
1	160.0	2000.0	51.33

2	160.0	1000.0	51.33
3	160.0	750.0	51.33
4	160.0	2000.0	51.33
5	160.0	750.0	51.33
6	160.0	2000.0	51.33
7	160.0	250.0	51.33
8	160.0	250.0	51.33
9	160.0	250.0	51.33

Different amounts of OD demand are loaded onto this network. The OD loading values are exactly the same as were for the DTA case (Chapter 4). The deviation of the seed OD demand value matrix is also the same as was for the DTA case. Three time intervals for OD demand loading are considered. Each time interval is of five minutes. The results are summarized in Table 5.5 (see page 54 for table layout details) below.

Table 5.5 Estimation results (Kalman Filter, network 4)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
Run 1				
75	75	0%	75.0	0%
150	135	10%	149.6	0.3%
75	60	20%	73.1	2.5%
105	90	14.3%	106.1	1.1%
135	120	11.1%	128.1	5.11%
90	75	16.67%	90.0	0%
Run 2				
195	180	7.7%	193.4	0.82%
180	165	8.33%	178.0	1.11%
165	150	9.1%	161.4	2.2%
150	135	10%	150.0	0%
165	150	9.1%	158.1	4.2%
180	165	8.33%	171.3	4.8%
Run 3				
300	345	15%	310.1	3.4%
450	480	6.67%	474.5	5.4%
300	330	10%	304.4	1.5%
375	330	12%	365.2	2.6%
225	270	20%	238.7	6.1%
300	270	10%	288.1	4.0%
Run 4				
450	525	16.67%	490.1	8.9%
525	600	14.3%	550.0	4.8%
450	525	16.67%	471.2	4.7%
525	600	14.3%	530.1	1.0%
570	675	18.4%	600.0	5.3%
450	540	20%	480.1	6.7%
Run 5				
600	675	12.5%	650.1	8.4%
675	750	11.1%	702.1	4.0%
630	720	14.3%	680.4	8.0%
660	750	13.6%	702.6	6.5%
720	825	14.6%	788.3	9.5%
690	780	13.1%	724.7	5.0%
Run 6				
750	870	16.0%	824.1	9.9%

870	975	12.1%	929.1	6.8%
900	975	8.33%	970.1	7.8%
825	930	12.7%	880.4	6.7%
930	975	4.8%	975.0	4.8%
870	975	12.1%	931.4	7.1%

Also, a comparison of these results is made with the results obtained from DTA (Table 4.2). The OD demand values over here (Table 5.5) are higher than the values of Table 4.2 because the values in Table 4.2 are for 20 seconds three time periods in 1 minute) where the values in Table 5.5 are for 5 minutes each which results in values in Table 4.2 to be multiplied by (3*5 =15). The comparison is summarized in Figure 5.3 below.

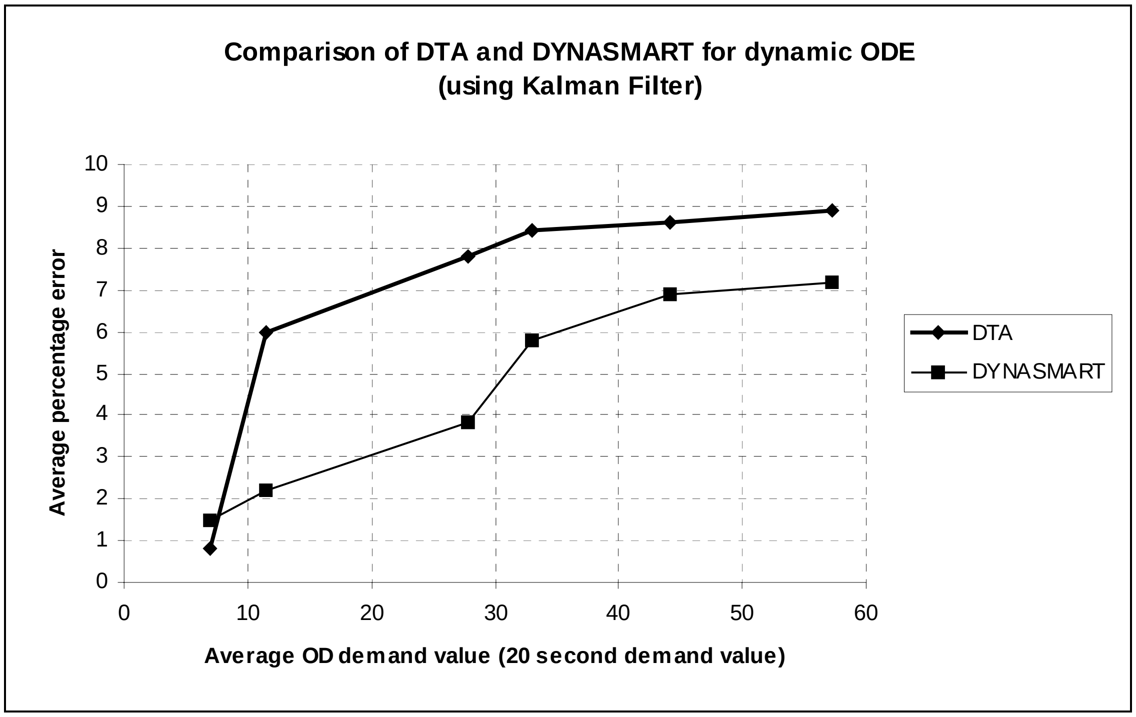


Figure 5.5 Comparison of DYNASMART and DTA for ODE using Kalman Filter

It can be seen from Figure 5.5 that the average percentage errors for DTA are higher for most of the OD demand values. Dynamic OD demand estimation using Kalman Filter and DYNASMART performs much better for most of the cases.

5.7 CONCEPTUAL FRAMEWORK FOR ODE USING LEAST SQUARES

As in the Kalman filter estimation scheme, the least squares formulation also relates the origin/destination demands and the section densities via the following equation:

$$y = Bq + v$$

The basic non-recursive least squares seeks to minimize the sum of squares errors between the actual section density and the estimated measurements $B\hat{q}$

$$\text{min. } Z = \text{min. } \left(y - B\hat{q}\right)' \left(y - B\hat{q}\right)$$

The resulting origin/destination demand is expressed as

$$\hat{q} = \left(B'B\right)^{-1} B'z$$

The recursive least squares computation procedure allows the incorporation of the additional current measurement of section density without redoing an entire Least Squares estimation over the previous time period and the current time period.

The network consists of a section of 405 North Freeway, Jeffrey and Sand Canyon. The network has four zones. Two zones are origin zones and the other two are destination zones. Origin zones have one link each as the demand generation link. The network is shown in Figure 5.2. The link characteristics are shown in Table 5.1. This is the same network as used for Kalman Filter tests. There are four OD pairs. OD pair 1 has zone 1 as the origin and zone 3 as the destination. OD pair 2 has zone 1 as the origin and zone 4 as the destination. OD pair 3 has zone 2 as the origin zone and zone 3 as the destination zone. OD pair 4 has zone 2 as the origin zone and zone 4 as the destination zone. The demand is loaded on the network in four time intervals of 5 minutes each.

The first case considered is that of a low OD demand loading onto the network. Low loading does not cause any traffic delays on the network. The results are summarized in Table 5.6 (see page 54 for table layout details) below.

Table 5.6 Estimation results (Least Squares, real network, low loading)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
60	50	16.67%	56.1	6.5%
80	70	12.5%	74.0	7.5%
80	70	12.5%	76.0	5.0%
50	60	20%	51.2	2.4%
60	50	16.67%	58.8	2.0%
70	60	14.3%	68.2	2.6%
80	70	12.5%	76.1	4.9%
50	60	20%	50.0	0%
60	50	16.67%	60.0	0%
50	60	20%	51.1	2.2%
60	60	0%	60.0	0%
80	80	0%	80.0	0%
40	50	25%	42.0	5.0%
60	80	33.3%	61.2	2.0%

70	80	14.3%	72.0	2.8%
75	75	0%	75.0	0%
<u>Run 2</u>				
60	50	16.67%	56.8	5.33%
80	60	25%	76.0	5.0%
80	60	25%	76.0	5.0%
50	80	60%	54.1	8.2%
60	40	33.3%	61.2	2.0%
70	50	28.6%	66.0	5.7%
80	50	37.5%	73.2	8.5%
50	70	40%	53.1	6.2%
60	80	33.3%	60.0	0%
50	80	60%	52.0	4.0%
60	80	33.3%	61.1	1.83%
80	50	37.5%	78.0	2.5%
40	60	50%	40.0	0%
60	80	33.3%	60.0	0%
70	50	28.6%	66.0	5.7%
75	50	33.3%	73.6	1.9%

The results indicate that the dynamic OD estimation was able to track the OD demand values to within ninety-six percentage (96%) accuracy when the historical OD demand values are deviated from the true OD demand values by an average of 14 percent. Also, when the average historical OD demand values are deviated by an order of 36 percent, the average OD demand values are estimated within ninety-five percentage (95%) accuracy.

The second case considered is that of a heavy loading of OD demand values on the network. Heavy loading leads to congestion and results in average traffic for a vehicle to traverse the network. The results for this case are summarized in Table 5.7 (see page 54 for table layout details) below.

Table 5.7 Estimation results (Least Squares, real network, heavy loading)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
Run 1				
155	135	12.9%	150.0	3.2%
188	188	0%	188.0	0%
188	148	21.3%	176.1	6.3%
167	147	11.9%	161.0	3.6%
256	256	0%	251.8	1.6%
137	137	0%	126.8	7.5%
147	157	6.8%	155.0	5.4%
27	27	0%	27.0	0%
136	116	14.7%	128.0	5.9%
129	139	7.8%	135.0	4.7%
238	218	8.4%	223.0	6.3%
119	109	8.4%	110.6	7.1%
156	156	0%	150.0	3.9%
119	119	0%	112.7	5.3%
138	138	0%	138.0	0%
129	129	0%	129.0	0%
Run 2				
155	205	35.5%	163.1	5.2%
188	220	17.1%	195.0	3.7%
188	150	20.2%	198.0	5.0%
167	200	19.8%	170.1	1.9%
256	200	28%	251.3	1.83%
137	160	16.8%	143.6	4.8%
147	180	22.4%	156.0	6.1%
27	75	177.7%	32.1	18.9%
136	150	10.3%	140.0	2.9%
129	150	16.3%	131.7	2.1%
238	200	16%	226.0	5.0%
119	150	26.1%	125.0	5.0%
156	120	23.1%	150.0	3.9%
119	150	26.1%	115.4	3.0%
138	150	8.7%	136.0	1.5%
129	150	16.3%	136.2	5.6%

The results show that the dynamic OD demand estimation was able to track the OD demand to within 95 percentage accuracy when the historical demand values are away from the true OD demand values by more than 5 percent. The average estimation accuracy drops to 93 percentage when the historical OD demand values are deviated from the true OD demand values by more than 30 percent.

A comparison of the two cases show that the average percentage error of estimation of dynamic OD demand values are more for the heavy loading of the network case (Figure 5.6). This is in conjunction with the fact that when a heavy demand is loaded onto the network, it results in traffic congestion which results in abnormal link densities and this leads to more errors in the tracking of true OD demand value.

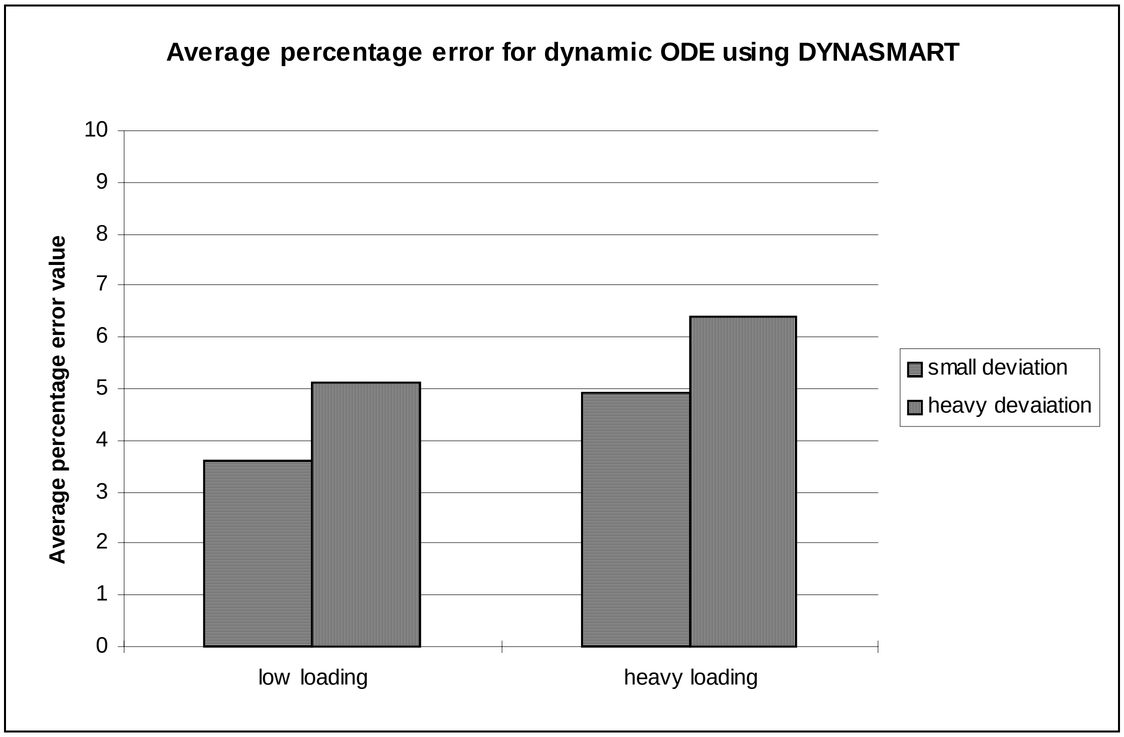


Figure 5.6 Comparison of low and heavy loading on the network (Least Squares, real network)

The other network that was used was composed of seven nodes, nine links and three zones. The link characteristics are shown in Table 5.4. OD pair 1 has zone 1 (node 1) as the origin zone and zone 3 (node 3) as the destination zone. OD pair 2 has zone 1 (node 1) as the origin zone and zone 2 (node 4) as the destination zone. A graphical representation of the network is shown in Figure 5.4. Table 5.8 (see page 54 for table layout details) gives the results obtained for the various cases.

Table 5.8 Estimation results (Least Squares, network 4)

True OD matrix	Historical OD (seed) matrix	%Error of historical matrix	Estimated OD matrix	%Error from true value
<u>Run 1</u>				
75	75	0%	75.0	0%
150	135	10%	150.0	0%
75	60	20%	75.0	0%
105	90	14.3%	106.1	1.0%
135	120	11.1%	135.0	0%
90	75	16.67%	90.1	0.11%
<u>Run 2</u>				
195	180	7.7%	194.1	0.5%
180	165	8.33%	180.0	0%
165	150	9.1%	163.2	1.1%
150	135	10%	150.0	0%
165	150	9.1%	160.0	3.0%
180	165	8.33%	175.0	2.8%
<u>Run 3</u>				
300	345	15%	300.0	0%
450	480	6.67%	460.1	2.2%
300	330	10%	310.3	3.4%
375	330	12%	360.5	3.9%
225	270	20%	230.8	2.6%
300	270	10%	295.4	1.5%
<u>Run 4</u>				
450	525	16.67%	470.2	4.5%
525	600	14.3%	550.0	4.7%
450	525	16.67%	460.1	2.2%
525	600	14.3%	535.7	2.0%
570	675	18.4%	575.8	1.0%
450	540	20%	470.0	4.4%
<u>Run 5</u>				
600	675	12.5%	625.0	4.2%
675	750	11.1%	700.1	3.7%
630	720	14.3%	630.0	0%
660	750	13.6%	700.3	6.1%
720	825	14.6%	754.6	4.8%
690	780	13.1%	721.8	4.6%
<u>Run 6</u>				
750	870	16.0%	800.0	6.67%

870	975	12.1%	900.7	3.5%
900	975	8.33%	940.1	4.5%
825	930	12.7%	860.3	4.3%
930	975	4.8%	960.0	3.2%
870	975	12.1%	904.5	4.0%

Also, a comparison of the results obtained using Kalman Filter and the Least Squares for estimating dynamic OD demand values using DYNASMART is made. Figure 5.7 summarizes the various percentage errors obtained. Least Squares perform better than Kalman Filter because Kalman Filter assumes auto-correlation of the state variables so there is a dependence on historical OD. While Least Squares is only indirectly dependent on historical OD via the "B" matrix.

It can be seen from figure 5.7 that the average percentage errors for the Least Squares method are smaller than the average percentage errors for the Kalman Filter method for the same cases. Least Squares method is able to estimate dynamic OD demand values closer to the true OD demand values. It can also be seen that the Least Squares perform better by an average of forty-eight percentage (48.0%). The average performance percentage is obtained by using the following formula:

$$Average \quad Performance \quad Percentage = \frac{\left(K_e - L_e\right) * 100}{K_e * N}$$

where,

K_e = Average percentage error for Kalman Filter method,

L_e = Average percentage error for Least Squares method,

N = Total number of cases.

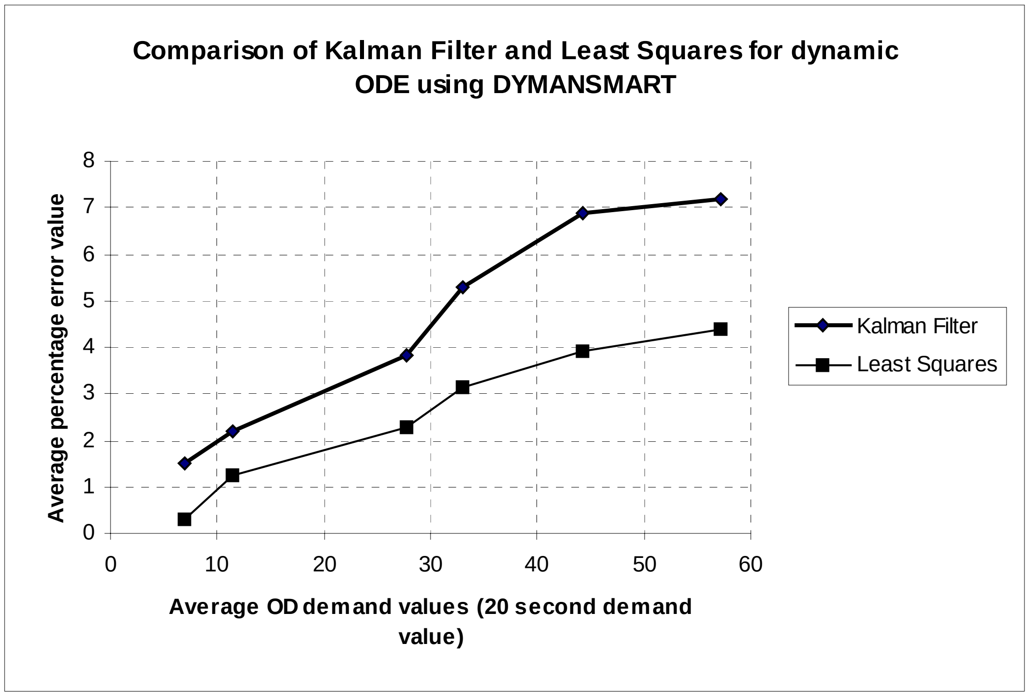


Figure 5.7 Comparison of Kalman Filter and Least Squares method for dynamic OD estimation using DYNASMART

5.8 COMPARISON OF DYNAMSART AND DTA USING KALMAN FILTER AND LEAST SQUARES METHODS

A comparison of both optimization methods (Kalman Filter and Least Squares) using both the traffic assignment algorithms (DTA and DYNASMART) is made. The results are summarized in Figure 5.8. These results are for network 1 (network 4 for the DYNASMART cases). Network 4 is a DYNASMART compatible version of network 1.

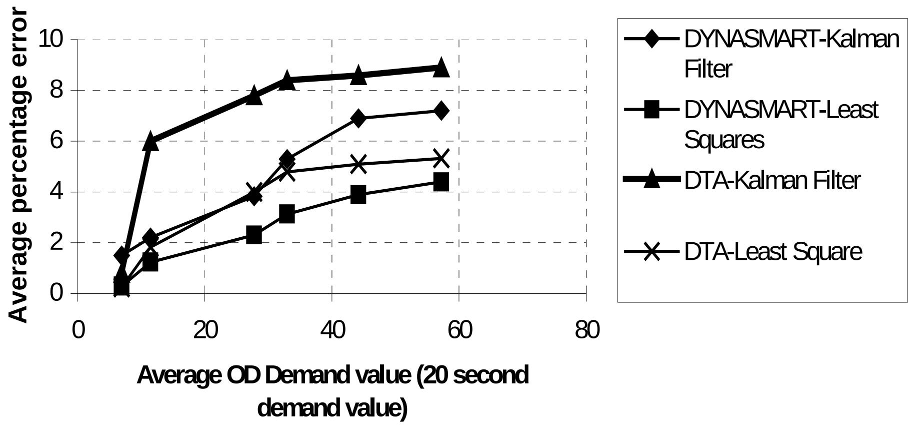


Figure 5.8 Comparison of all the traffic assignment methods for all the optimization techniques (Kalman Filter and Least Squares)

Figure 5.8 shows that the best dynamic OD demand value estimations are made by a combination of DYNASMART and Least Squares methods. The worst results were obtained for the combination of DTA and Kalman Filter. The average percentage performance for the combination of DYNASMART and Least Squares is sixty-four percentage (64.4%) over the combination of DTA and Kalman Filter. Also, the combination of DYNASMART and Least Squares was able to predict the dynamic OD demand values to within ninety-five percentage accuracy (95%).

CHAPTER 6. SENSITIVITY ANALYSIS

6.1 INTRODUCTION

For the dynamic Origin/destination demand estimation problem, sensitivity analysis (SA) was done by perturbing a number of variables one by one and studying their impact on the results of the estimation framework. The OD estimation framework that is used is :

Least Squares	For minimizing the errors between the actual and the estimated section densities.
DYNASMART	For the assignment of the evaluated OD matrix onto the network

This framework was chosen because this framework gave the best results (minimum errors) for dynamic OD estimation. The variable that is perturbed is given below:

- Perturbing the Saturation Volume on a link.

The effects of the perturbations of these variables are presented in the following sections.

6.2 PERTURBING THE SATURATION VOLUME ON A LINK

The network chosen for this sensitivity analysis is the smaller network. The network has seven nodes, nine links and three zones. A graphical representation of the network is shown in Figure 5.4. The link characteristics are shown in Table 5.4. For sensitivity analysis, the Saturation volume on link number two (link from node 1 to node 3) was decreased by twenty percentage (20%) resulting in a drop in the Saturation volume from 1800 vehicles per hour to 1440 vehicles per hour. The Saturation volume was then increased by twenty percentage (20%) resulting in an increase from 1800 vehicle per hour to 2160 vehicles per hour. Link number two was chosen as this was the link which had the maximum density for most of the test runs done earlier. The results of this perturbation in the link saturation volume are shown in Figure 6.1.

From Figure 6.1 it can be seen that the average percentage errors have gone up slightly for all the cases when the Saturation volume is decreased. This is due to the fact that the reduction in saturation volume on a link diverts most of the traffic onto the other links (path). This, in general, causes more congestion on the other links of the network (because every link has a limit on the saturation volume that it can handle) and hence it becomes more difficult to trace the dynamic origin/destination demands. The average percentage errors did not change for cases where the OD demand was low because a decrease in saturation volume does not affect much if the number of vehicles on the network are lesser in number. There are no congestion and each vehicle travels at pretty much the free flow speed. Hence, there is not much difficulty in tracing the dynamic OD demand values. The increase in saturation volume causes slight reduction in average percentage errors for larger OD demand values. The affect is not seen for lower demand values. The average percentage errors decrease for larger OD demand values is due to the fact that increasing the saturation volume on a link causes more vehicles to be accommodated on the link resulting in lesser congestion and hence making it easier to trace dynamic OD demand values.

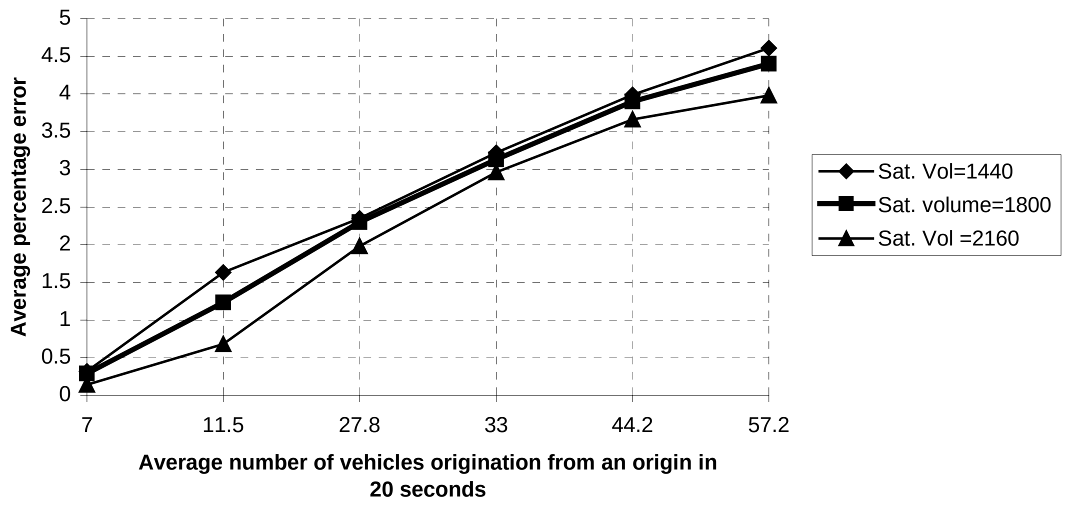


Figure 6.1 Effects of perturbing the saturation Volume on a link

The plot of average travel time (Figure 6.2) for a vehicle to traverse the network shows an increase in the travel times for the cases when the saturation volume was decreased. This increase is more for higher average OD demand values. This is because of the fact that higher OD demand generates more vehicles on the network and these more vehicles cause even more congestion on the network. This leads to larger travel times. Increasing the saturation volume on a link causes a slight decrease in travel times for higher OD demand values. Increasing the saturation volume enables the link to accommodate more vehicles on it without causing congestion and hence lower travel times are observed.

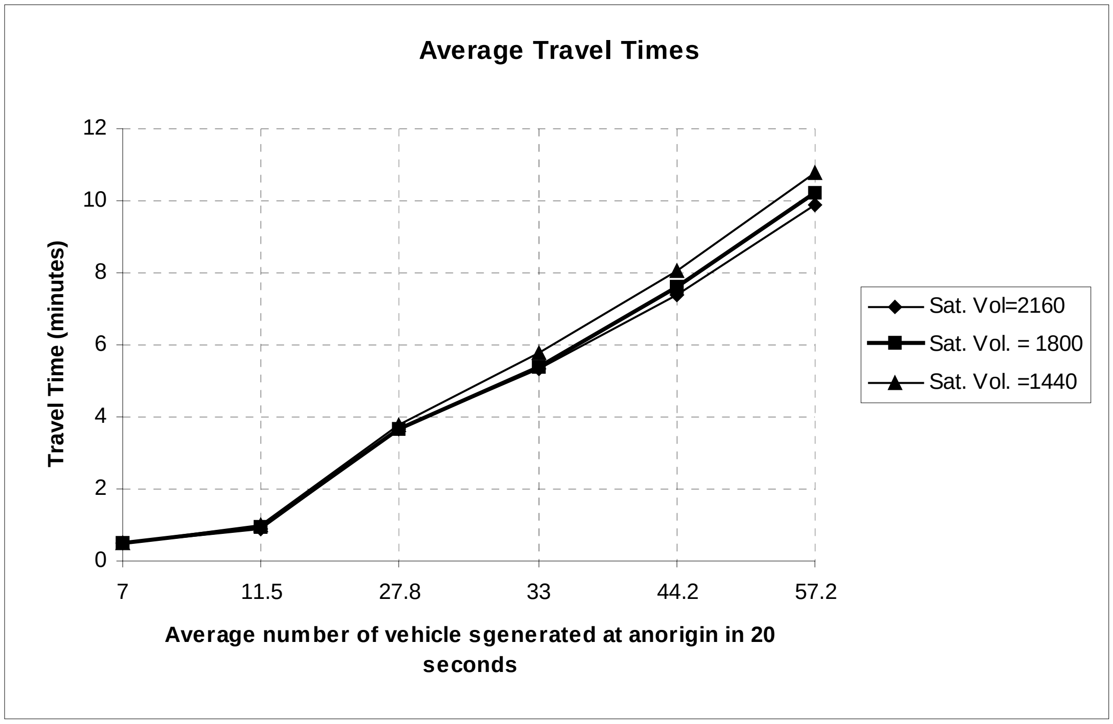


Figure 6.2 Effect of perturbing the Saturation Volume on Average Travel time

The time varying results of perturbing the saturation volume on link number two for all the scenarios (increasing, decreasing saturation volumes) and its effect on all the other links are shown in Table 6.1

Table 6.1 Sensitivity Analysis for dynamic OD estimation based on Link Loads.

Link	Time Step	Saturation Volume = 1800 vehicles/hour			Saturation Volume = 2160vehicles/hour			Saturation Volume = 1440 vehicles/hour		
		True OD	Est. OD	% Diff.	True OD	Est. OD	% Diff.	True OD	Est. OD	% Diff.
1	1	32	31	3.125	32	32	0	32	31	3.13
	2	27	27	0	27	28	-3.7	27	27	0
	3	27	25	7.41	27	28	-3.7	27	28	-3.7
	4	29	36	-24.1	29	35	-20.7	29	35	-20.7
	5	38	31	18.42	38	21	44.7	38	31	18.4
	6	25	26	-4.0	25	25	0	25	24	4.0
	7	35	37	-5.71	35	37	-5.7	36	37	-2.78
	8	40	35	12.5	40	39	2.5	38	39	-2.6
	9	31	28	9.68	31	24	22.6	32	26	18.8
	10	30	35	-16.7	22	33	-50.0	23	40	-73.9
2	1	70	72	-2.86	70	73	-4.3	70	72	-2.9
	2	70	73	-4.3	70	74	-5.7	68	72	-5.9
	3	70	69	1.43	70	68	2.9	67	65	2.9
	4	71	63	11.26	75	62	17.3	72	61	15.3
	5	60	64	-6.67	60	68	-13.3	60	64	-6.67
	6	66	72	-9.1	61	69	-13.1	58	66	-13.8
	7	68	62	8.83	68	62	8.82	68	63	7.4
	8	59	64	-8.48	59	68	-18.6	59	62	-5.1
	9	69	70	-1.45	69	69	0	70	68	2.9
	10	64	69	-7.81	65	66	-1.54	40	62	-55.0
3	1	9	8	11.1	9	8	11.1	9	8	11.1
	2	9	12	-33.3	6	9	-50.0	6	12	-100
	3	8	8	0	8	11	-37.5	8	11	-37.5
	4	14	16	-14.3	14	16	-14.3	14	16	-14.3
	5	18	14	22.2	18	9	50.0	18	14	22.2
	6	8	8	0	8	13	-62.5	8	8	0
	7	14	18	-22.2	15	15	0	15	18	-20.0
	8	11	16	-45.5	11	14	-27.3	14	16	-14.3
	9	17	12	29.4	17	8	52.9	15	12	20
	10	9	10	-10.0	9	12	-33.3	11	13	-18.2
4	1	26	26	0	26	25	3.85	26	26	0
	2	30	29	3.3	30	29	3.33	30	29	3.33
	3	22	33	-50.0	22	31	-40.9	22	33	-50
	4	24	16	33.3	24	16	33.3	24	16	33.3
	5	25	18	28.0	25	23	8.0	26	18	30.8
	6	16	23	-43.8	16	31	-93.8	16	23	-43.8
	7	23	22	4.35	23	29	-26.1	19	22	-15.8

	8	25	19	24.0	25	25	0	23	18	21.7
	9	26	19	26.9	26	23	11.5	26	19	26.9
	10	18	29	-61.1	14	23	-64.3	16	26	-62.5
	1	9	9	0	9	9	0	9	9	0
	2	14	21	-50.0	14	20	-42.9	13	21	-61.5
	3	9	7	22.2	9	7	22.2	9	7	22.2
	4	9	16	-77.7	9	15	-66.7	9	15	-66.7
5	5	12	13	-8.33	12	10	16.7	12	12	0
	6	7	12	-71.4	9	17	-88.9	7	11	-57.1
	7	11	7	36.4	11	13	-18.2	9	6	33.3
	8	7	10	-42.9	7	10	-42.9	8	8	0
	9	16	5	68.8	16	11	31.3	5	5	58.3
	10	9	10	-11.1	9	9	0	10	10	-42.9
	1	42	41	2.4	42	41	2.4	42	41	2.4
	2	49	37	24.5	49	37	24.5	48	37	22.9
	3	41	45	-9.8	41	47	-14.6	41	48	-17.1
	4	46	39	15.2	46	41	10.9	46	41	10.9
6	5	44	40	9.1	44	46	-4.5	44	39	11.4
	6	44	51	-15.9	44	36	18.2	44	53	-20.5
	7	37	48	-29.7	37	39	-5.4	45	52	-15.6
	8	45	47	-4.4	45	36	20.0	40	47	-17.5
	9	39	41	-5.1	39	46	-17.9	39	41	-5.12
	10	39	38	2.56	39	47	-20.5	35	35	0
Average Absolute Percentage Error				21.83			22.05			22.3

The estimated OD matrix that has been used here is the OD matrix that resulted from dynamic OD estimation using DYNASMART and Least Squares method. A plot of the time variation of the total link loads is shown in Figure 6.3. It can be seen from this plot that the effect of decreasing the saturation volume is more prominent that the effect of increasing the saturation volume on a link. Also, the effect of saturation volume decrease results in the decrease of the link loads. This decrease starts to happen as early as the second time step where as the increase in link load as a result of increase in saturation volume on a link starts from time step 4. It can also be seen that the effect of decreasing saturation volume over a link results in overall decrease in link loads which eventually results in a decrease in the dynamic OD demand matrix (as can be seen in Table 6.1).

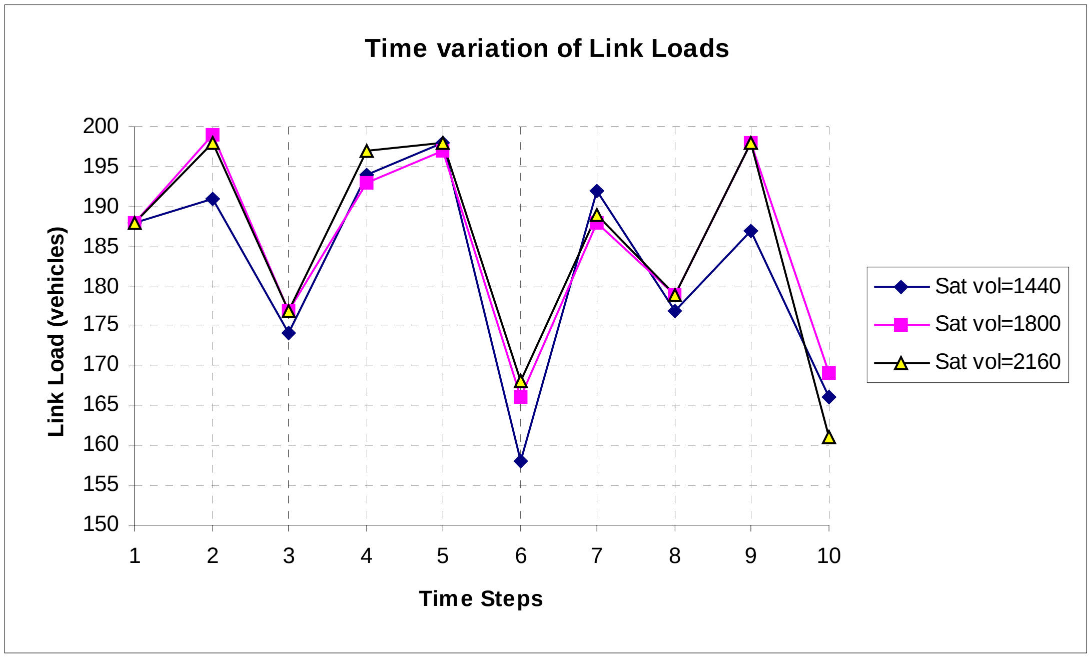


Figure 6.3 Time variation of link loads resulting from loading of True OD demand matrix on the network.

A time varying comparison of the link loads for the estimated dynamic OD demand matrices are shown in Figure 6.4. It can be seen from Figure 6.4 that the overall effect of the increase in saturation volume of a link results in the overall increase in the link loads.

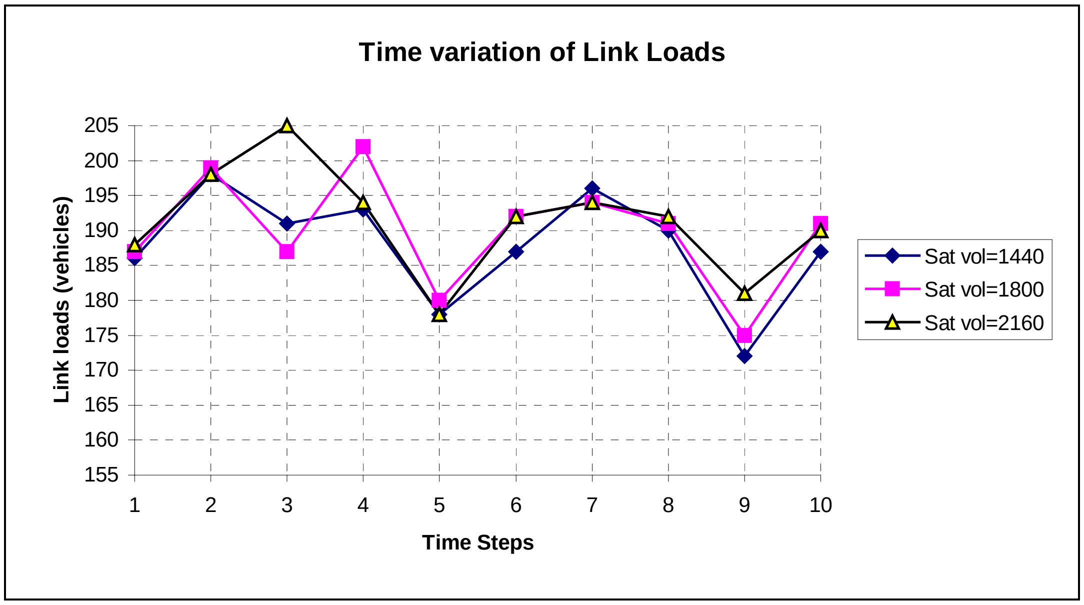


Figure 6.4 Time variation of link loads resulting from estimated dynamic OD demand matrices.

CHAPTER 7. CONCLUSIONS AND FUTURE RESEARCH

7.1 SUMMARY AND CONCLUSIONS

This project has demonstrated the basic feasibility of analyzing the potential of benefits of using section density as a measure for the estimation of dynamic origin/destination demands. Realistic simulations have been performed utilizing network and demand levels based on the Irvine testbed network. Two traffic assignment schemes were utilized. One based on simulation (DYNASMART) and the other an analytical scheme (DTA). Also, two different optimization methodologies (Kalman Filter and Least Squares) were utilized to minimize the errors between the predicted and the true section measures.

The true origin/destination demand is loaded on the network and the section measures (density) is recorded on each link in the OD-link incidence matrix (the “B” matrix). Then the perturbed origin/destination demand (seed matrix) is loaded and the resulting section measures (recorded in the new “A” matrix) are compared with the section measures obtained from the true origin/destination demand matrix. Optimization algorithms (Kalman Filter, Least Squares) are used iteratively to minimize the errors between these section measures obtained from the use of different origin/destination demand matrices.

Results of the dynamic origin/destination demand estimation suggest achievable benefits of up to ninety-four percent accuracy in the estimation of dynamic O/D matrix. Possibly greater benefits may occur under the scenarios of smaller networks or the seed origin/destination demand matrix being close to the true origin/destination demand matrix. The sensitivity analysis suggests that decreasing the saturation volume on a link have an overall affect of redistributing the demands to the links with more volume capacities and thus affecting the dynamic origin/destination demand estimation. This is due to resulting congestion on links making it difficult to trace back the demand values.

7.2 GENETIC ALGORITH APPROACH TO DYNAMIC OD ESTIMATION

The field of Genetic algorithms (GA) has been growing since the early 1970s, but only recently has it been increasingly applied to real-world problems. GAs were developed by John Holland of the University of Michigan. Genetic algorithms are search and optimization procedure motivated by the evolution in the natural world, which is controlled by the process of natural selection. Organisms most suited for their environment tend to live long enough to reproduce and pass-on their beneficial and survival enhancing characters to their children, whereas less-suited organisms do not survive for long and often die before producing children or produce fewer and/or weaker children with lower and lower chances of survival. In this procedure, Darwin’s survival-of-the-fittest principle is applied iteratively on a population of parent chromosomes (strings, representing the problem parameters) so as to produce better children as new generations evolve.

In the real world, an organism's characteristics are encoded in its DNA. GAs store the characteristics of artificial organisms in electronic chromosomes. Decision variables are usually mapped and represented by a string (chromosome) of binary alphabets (genes). For problems with more than one decision variable, each variable is usually represented by a sub-string. All sub-strings are then concatenated together to form a bigger string. For example, if a problem consists of two decision variables, x_1 and x_2, and if five and seven bits binary coding are chosen for first and second variables, respectively, then the combined string is as follows:

$$\left(x_1, x_2\right) \equiv 10110|1001011$$

To retrieve x_1 and x_2 values, each sub-string is first decoded and then mapped into the desired range. For example, if the minimum and maximum possible values for x_1 is 0.0 and 20.0, respectively, the foregoing string has an x_1 value equal to

$x_1 = 0.0 + (20.0 - 0.0) \times 22/31 = 14.18$, since the decoded value of the binary string is (10110) is 22. The maximum decoded value of the variable x_1 is $2^5 - 1$ or 31. Similarly, the value of the second variable can also be calculated.

Figure 7.1 presents a pseudo-code of the working of a simple GA. First, a population of strings representing the decision variables is created at random using a random number generator. The size of the population depends on the string length and the problem being optimized. Each string is then evaluated. The evaluation process requires decoding of the decision variables from the string and then using the decoded values to calculate the objective function value, which is used as a measure of the “goodness” of the string. In GA terminology, the objective function value of a string is known as the fitness of the string. Once all strings are evaluated, three main genetic operators – *reproduction, crossover,* and *mutation* – are used to create new population.

```
Initialize a population of strings at random
Evaluate each string in the population
   repeat
         Reproduction
         Crossover
         Mutation
         Evaluation of the population
   until  (termination criterion)
```

Figure 7.1 Working of Genetic Algorithms

Reproduction (or selection) is an operator that makes more copies of better strings in a population. Among many reproduction schemes, proportionate selection is most commonly used. In the proportionate selection, a string is selected with a probability ${}^{f_i}\!/_{f_{avg}}$, where f_i is the fitness of the i th string and f_{avg} is the average fitness of all strings in the population. This indicates that a string with higher fitness value has a higher probability of getting selected than a string with comparatively lower fitness value. Different selection schemes vary in principle by introducing different number of copies of better strings into the population but, in all selection schemes the essential idea is that more copies of the string with higher fitness value are introduced.

After the reproduction phase is over, the population is enriched with good strings. Reproduction makes clones of good strings, but does not create any new string whereas, a *crossover* operator is used to recombine two strings (resulting after reproduction phase) with the hope of creating a better string. To preserve some of the good strings found previously, crossover is usually performed with a probability. Crossover recombines the genetic material in two parent chromosomes to make two children. One-point crossover occurs when parts of two parent chromosomes are swapped after a randomly selected point. For example, the action of a one-point crossover operator on two five bit strings is :

$$\begin{array}{r|l} 00 & 000 \\ 11 & 111 \end{array} \quad \Rightarrow \quad \begin{array}{r|l} 00 & 111 \\ 11 & 000 \end{array}$$

Two strings are chosen at random for a crossover. A crossing site (represented by the vertical line) is chosen at random. The contents of the right side of the crossing site are swapped between two strings. A number of variations to this crossover exists, but the essential idea is to exchange bits (information) between two good strings to obtain a string that is possibly better than the parents.

Another operator, *mutation*, is used to create diversity in the children. Under the action of this operator, a 1 changes to a 0 and vice versa with a small probability. Mutations also creates a new string, but its effect is considered secondary. It introduces diversity in the population whenever the population tends to become homogenous due to iterative use of selection and crossover operators. Applying cross-over to two identical strings produce an identical child. The only way to introduce variation in this case is through mutation.

After new strings are created, they are evaluated by decoding and calculating their objective function values (fitness). This completes a cycle of genetic algorithm (a cycle of GAs is also known as a *generation*). Such iterations are continued until termination criterion is satisfied.

The decision variable for dynamic OD estimation are the time-dependent origin/destination demand values. These decision variables will be coded in binary digits to form the binary string (gene). This binary coding can be done using some random number generator. Each of these binary decision variable (genes) then combine to form the binary string (chromosome). This binary string is then decoded to arrive at the actual value. These values will then be assigned onto the given network. The resulting link densities will then be compared to the actual densities and the error between the two link densities will be minimized. The fitness function will be an error minimizing function. The dynamic OD demand matrix that gives the best fitness will be the best dynamic OD for the given case at hand. The problem formulation is as follows:

1. Decide on the number of bits to represent each dynamic OD value.
2. Define the fitness function as an error minimizing function between the values of estimated and actual dynamic OD demand values.
3. Use random number generator to code each binary string (genes).
4. Create a big enough search space (typically around 300-500 chromosomes).
5. Combine these genes to form the chromosome.
6. Do reproduction, crossover and mutation.
7. Decode the resulting chromosomes.
8. Do traffic assignment for these dynamic OD values using any good traffic assignment model.
9. Compare the link density values with the actual (real) link density values.
10. Evaluate the fitness of each chromosome.
11. Keep the better chromosomes.
12. Repeat steps 6 through 11 until termination criterion is met.

There are however, some issues that should be dealt with carefully while defining the problem and formulating it. This approach may work very efficiently for smaller networks but, for bigger network this approach is time consuming. The bigger the network, the more the number of OD pairs and time steps involved. This results in a bigger chromosome which necessitates a bigger search space. The bigger chromosome and bigger search space results in more computation time. But, this problem can be overcome if some time series or forced learning technique like neural networks can be incorporated along with genetic algorithms.

The advantages of using GAs is that they are very powerful method for searching through a large and complex solution space featuring a large extent of local minima. GAs have immunity to local minima and are able to find global optimal solutions quickly. Unlike many traditional methods, GAs work with a population of points. This increases the possibility of obtaining global optimal solution even in ill-behaved problems. GAs use probabilistic transition rules, instead of fixed rules. This randomness in GA operators make the search unbiased toward any particular region in the search space. These advantages make GAs make another distinct choice for dynamic OD demand estimation.

7.3 INTRODUCTION TO NEURAL NETWORKS

Human brain can be viewed as a highly inter-connected network of relatively simple processing elements. These simple processing elements are biological neurons. The basic function of a biological neuron is to add up its inputs and produce an output if this sum is greater than some value, known as the *threshold value*. The inputs to the neuron arrive along the dendrites, which are connected to the output from other neurons by specialized junctions called *synapses*. These junctions alter the effectiveness with which the signal is transmitted; some synapses are good junction, and pass a large signal across, whilst others are very poor, and allow very little through. The cell body of a neuron receives all these inputs, and fires, if the total input exceeds the threshold value. The efficiency of the synapses at coupling the incoming signal into the cell body can be modeled by having a multiplicative factor on each of the inputs to the neuron. A more efficient synapse, which transmits more of the signal, has a correspondingly higher weight, whilst a weak synapse has a small weight. Figure 7.2 shows the basic model of a neuron.

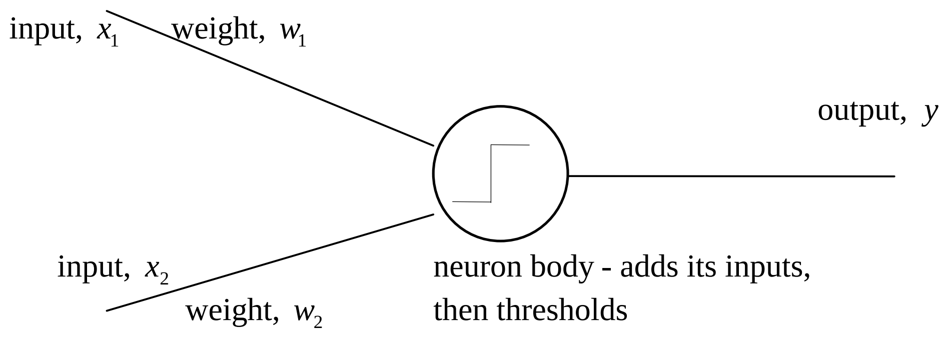


Figure 7.2 Basic model of a neuron

The body of the neuron performs a weighted sum of its inputs, compares this to some internal threshold level, and turns on only if this level is exceeded. If not, it stays off. This neuron can be trained to always produce correct outputs. The learning rule for a single neuron can be summarized as follows :

- set the weights and thresholds randomly
- present an input
- calculate the actual output by taking the thresholded value of the weighted sum of the inputs.
- alter the weights to reinforce correct decisions and discourage incorrect decisions — i.e. *reduce* the error
- present the next input etc.

Connecting these neurons together gives a neural network which needs to be trained in order to give correct results. It is this ability of these networks to learn that makes them especially useful for prediction purposes. Thus, artificial neural networks are mathematical systems that are comprised of a number of simple processing units that are linked via weighted interconnections. A processing unit is essentially an equation which is often referred to as a “transfer function”. A processing unit takes weighted signals from other neurons, combines them, transforms them and outputs a result. Neural networks are trained, meaning they use previous examples to establish (learn) the relationships between the input variables and their effect on the output by setting these weights. Once these relationships are established (the neural network is trained), the neural network can be presented with new input variables to generate predictions. The behavior of neural networks and how they map input data to output data is influenced primarily by the transfer functions of neurons, how the neurons are interconnected, and the values of the weights of those interconnections.

The basic neuron model is not good for solving complex problems and hence modifications are needed. The modified model is called *multi-layer perceptron*. This model has at least three layers; an input layer, an output layer, and a layer in between Figure 7.3.

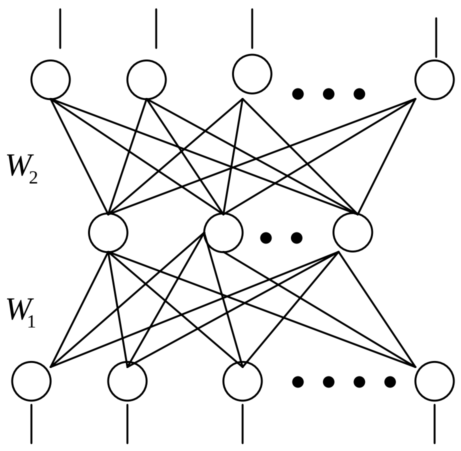


Figure 7.3 Architecture of a multi-layer neural network

Figure 7.3 illustrates the architecture of a multi-layer neural network. It consist of an input of *K* neurons, a hidden-layer of *M* neurons, and an output layer of *N* neurons. The input layer and the hidden layer are connected by a set of weights W_1, and the hidden layer and output layer are connected by a set of weights W_2. The input-output relationship of this neural network can be described by the following equation :

$$Y = \Phi\{W_2\Phi(W_1U + \theta_1) + \theta_2\}$$

where :

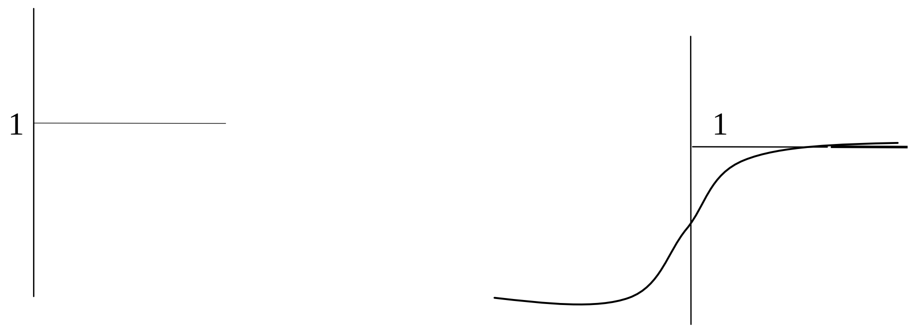
Φ = non-linear operator, normally a sigmoid or hyper-tangent type function such as

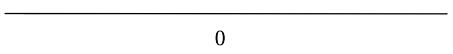
$\Phi(x) = \alpha / [1 + \exp(-x)]$

U = input vector;

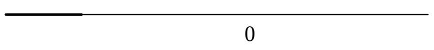
θ_1, θ_2 = vectors of thresholds for first hidden layer, second hidden layer, and output layer, respectively.

Because of the operator's non-linearity and the network's multi-layer structure, this type of network can perform highly non-linear mappings. Each unit in the hidden layer is like a perceptron unit, except that the threshold function is a sigmoid function unlike the step function (Figure 7.4) as was for the basic model. The units in the input layer serve to distribute the values they receive to the next layer, and so do not perform a weighted sum or threshold. Training patterns are presented to the network and the actual response is calculated. Comparison of the actual to the desired response enables the weights to be altered in proportion to their contribution to the error so that the network can produce a more accurate output the next iteration.





Step function



Sigmoidal threshold

Figure 7.4 Threshold functions

This model is also known as “back propagation” network as the error values are back propagated from one layer to the previous layer. Back propagating the error values through the network allow the weights between all the layers to be correctly adjusted. In this way the errors are reduced and the network learns.

However, a major limitation of the back-propagation algorithm is that it can only learn an input-output mapping that is *static*. This form of mapping is well suited for cases where both the input vector and the output vector represent *spatial* patterns that are independent of time. It can be used to perform nonlinear predictions on a stationary time series i.e., when its statistics do not change with time. One way in which this requirement can be accomplished is to introduce *time delays* into the synaptic structure of the network and to adjust their values during the learning phase. This gives us Time Delay Neural Networks (TDNN).

TDNN is a multilayer feed-forward network whose hidden neurons and output neurons are replicated across time. It is a more general form of back propagation. It employ back propagation technique for setting weights between neurons. It is like a back propagation network where there are multiple connections from the input neuron to the hidden neurons and from the hidden neurons to the output neurons. Each of these set of connections look back over time and sets its weight for each connection to minimize the mean error of the overall network. Figure 7.5 depicts such a network.

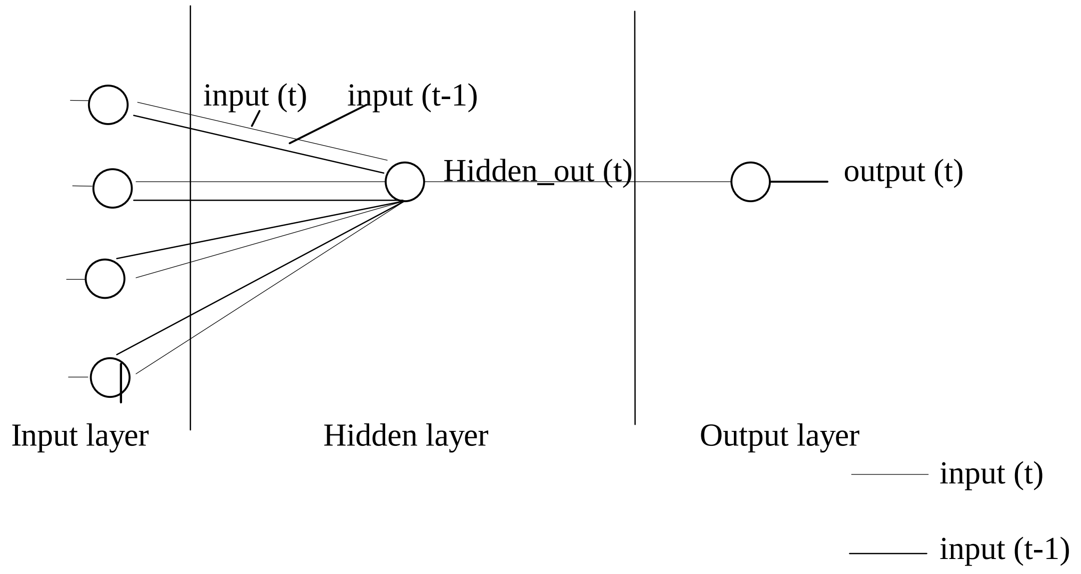


Figure 7.5 Architecture of a Time Delay Neural Network (TDNN)

Thus, TDNN can be thought of as a back propagation network with fixed time delays back N periods of time, like lagging the inputs by N periods of time. The one major difference is that the TDNN also does this with hidden neuron outputs too, thus seeing, remembering and using features in the data over time.

Dynamic OD demand is a function of time and so, TDNN is appropriate for problem formulation. We need to find a dynamic OD matrix that corresponds to the observed link densities at any given instant of time. Hence, the TDNN should be trained on the link densities as input vector (Figure 7.6). The output of this vector should be the OD demand matrix (Figure 7.6). The error function should try to minimize the errors between the predicted OD matrix and the actual OD matrix during the training process. The training of neural network should be done on a sufficiently large amount of data so that most of the observed cases could be covered. This will facilitate better estimation of the dynamic OD matrix.

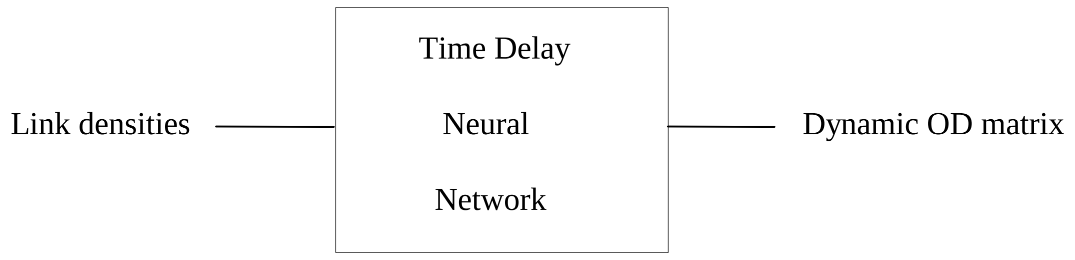


Figure 7.6 Conceptual model for ODE using neural networks

Once, the neural network has been trained for a sufficient amount of data, it can then be used for dynamic OD estimation. The input that required would be the observed link densities at any given instant of time and then the already trained neural network will estimate the dynamic OD to which the given link densities correspond to.

7.4 ROLLING HORIZON FRAMEWORK FOR DYNAMIC OD ESTIMATION

It can be seen that the long term predictions of future conditions is not reliable, a rolling horizon approach is used to constantly update present conditions and to forecast the future conditions. Using a rolling horizon formulation for dynamic OD matrix estimation with Kalman Filter or recursive least squares fits nicely, since they both allow incremental updating of new information.

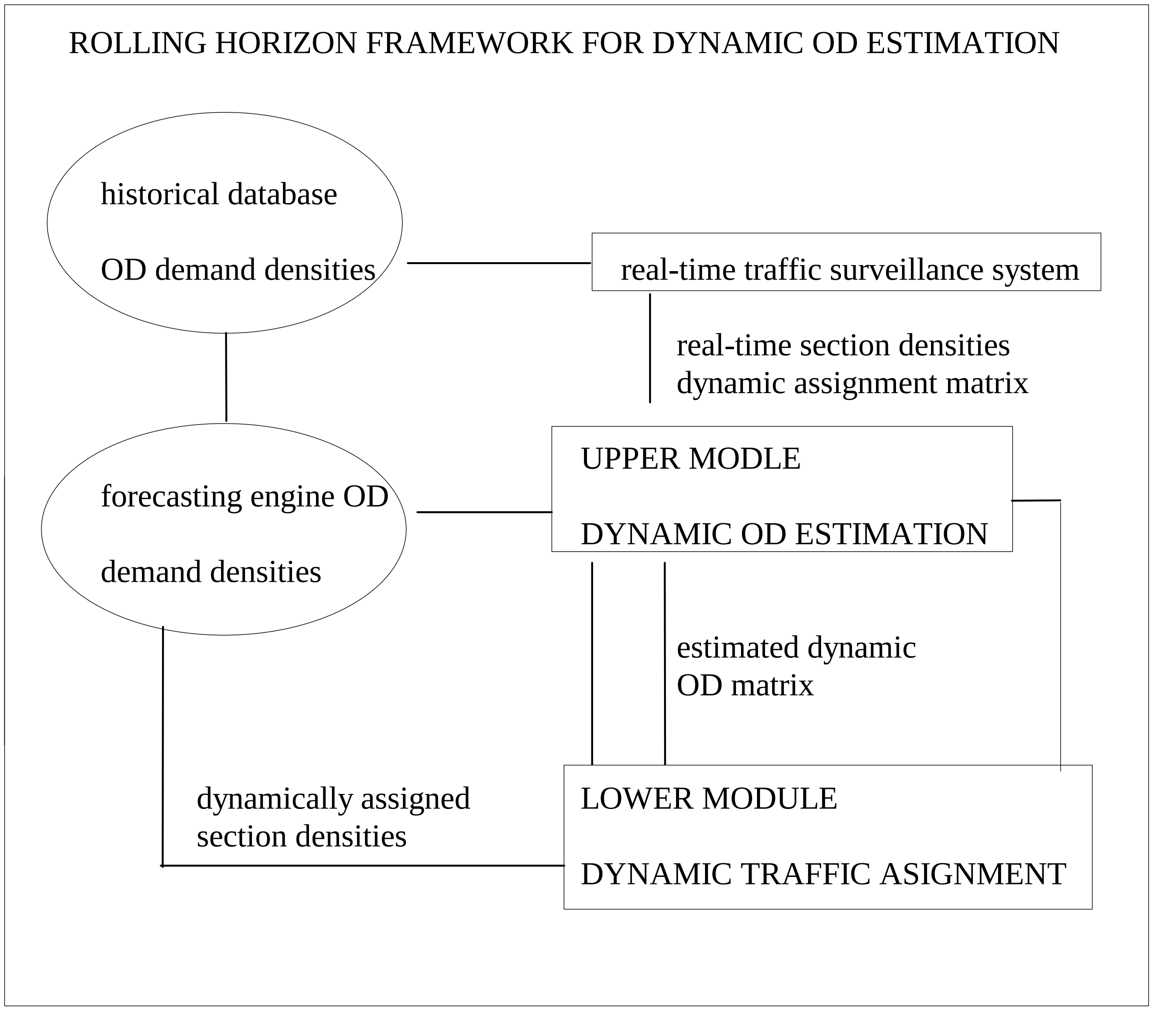


Figure 7.7 Flowchart of Rolling Horizon Dynamic OD Matrix Estimation/Traffic Assignment Framework

Figure 7.7 shows all the necessary components for a complete origin/destination demand estimation and dynamic traffic assignment framework. The forecasting module allows the incorporation of historical and current data in a discounted fashion. Thus the predicted values are based on historical values and is discounted according to a decaying exponential function (Koutsopoulos and Xu, 1992).

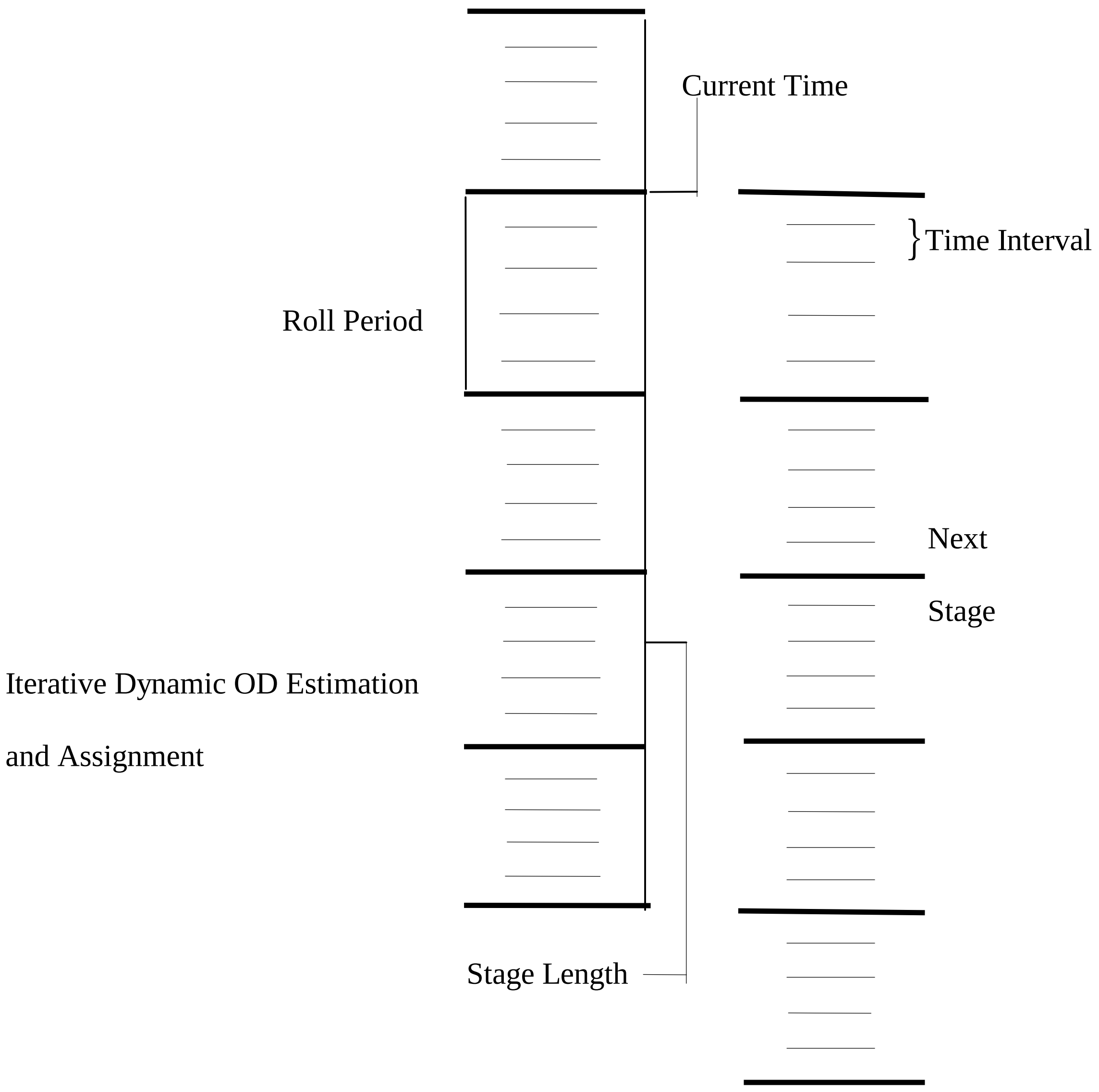


Figure 7.8 Rolling Horizon Dynamic OD Demand Estimation Framework

The general concept of using rolling horizon for dynamic OD demand estimation is illustrated in Figure 7.8 Each stage is divided into smaller time intervals, for example one minute demand intervals. Furthermore, each time interval can have subintervals of say 15 seconds that correspond to dynamic assignment intervals or the sampling interval for the link densities. The roll period is the amount of time during which reliable link densities can be obtained, for example a five minute period. During the roll period, an origin demand generation consistency check is performed, and the OD demands are corrected. The length of the roll period involves trade-offs between computation requirements and accuracy. If smaller periods are used, then more accuracy is obtained at the expense of more frequent computations. The stage length is the time corresponding to a solution of the OD demand estimation problem. The stage is based on a combination of short-term and less reliable mid-term forecast. An example of stage length could be thirty minutes of which twenty-five minutes will be using historical OD demand coupled with auto-regressive estimate of the OD demand matrix.

The hope in any transportation research is the eventual application of the research in the real life which would result in the improvement of the transportation system in relationship to its users. In view of this fact, real data, as opposed to simulated data, are critical in the realization of research ideas for real life implementation. The difficulty in getting real data is that the vehicles need to be tracked over both time and space, and only very expensive and complex systems exists for the recording of such data. The few feasible methods include time series aerial photography (from planes or tall buildings), re-correlated multi-station video tracking, and probe vehicles using tracking or AVI (Automatic Vehicle Identification) technology.

Faced with the problem of unavailability of such real-time data, some researchers have used real count data without validating origin/destination ground truthing matrix. For example, Van der Zijpp and Hamerslag (1994) used 5 minute period data for a 11 km belt-way in Amsterdam. The lack of a ground truth, however, also means that there is no way to access how close a particular formulation is able to replicate reality.

Because of the lack of such real-time data, some performance measures would need to be devised that would evaluate the proposed framework effectively and fairly. A consistency check can be applied by using a modified Kirchoff's constraint. The estimated demands would then need to be approximately equal to the sum/difference of the section densities that have the origin as their head or tail nodes. This is only applicable in the case of a sparse network where the origins are spread apart. In the more general case, the estimated demands would still need to be smaller than the aforementioned sum/difference. In other words, for an estimated demand $\hat{q}(i, l)$ from origin i at time l and section densities originating/departing from node n at time k

$$\hat{q}(i, l) \leq \sum_{m_o \in N_o(n)} y(k, m_o) - \sum_{m_i \in N_i(n)} y(k, m_i) \quad \forall \quad i \in N_i, l \in N_l, k = l, i = n$$

Here, m_o and m_i designate the links that enter/exit node n.

Another criterion that is common to numerical optimization is the time to convergence. This can be a measure of the overall convergence or of a percentage of the origin demands. Both single step and rolling horizon frameworks can be subjected to such criterion.

REFERENCES

Ashok, K., and Ben-Akiva, M.E. (1993) Dynamic Origin-Destination Matrix Estimation and Prediction for Real-Time Traffic Management Systems. Transportation and Traffic Theory, Elsevier Science Publishers B.V., pp. 465-484.

Bell, M.G.H., (1991) The Real Time Estimation of Origin-Destination Flows in the Presence of Platoon Dispersion. Transportation Research B, Vol. 25, Nos. 2/3, pp. 115-125.

Bell, M.G.H., and Grosso, S., (1998) The Path Flow Estimator as a Network Observer. Traffic Engineering + Control, October 1998, pp. 540-549.

Bell, M.G.H., Shield, C.M., Busch, F., and Kruse, Gunter (1995) A Stochastic User Equilibrium Path Flow Estimator. Submitted to Transportation Research C.

Camus, R., Cantarella, G.E., and Inaudi, D. (1994) O-D Prediction for Real-Time Traffic Management. IFAC Transportation Systems, Tirujin PRC, pp. 707-711.

Chakroborty, P., Deb, K., and Subrahmanyam (1995) Optimal Scheduling of Urban Transit Systems using Genetic Algorithms. Journal of Transportation Engineering, Nov./Dec. issue, pp. 544-553.

Chan, G. and Wu, J. (1994) recursive Estimation of Time-Varying Origin-Destination Flows From Traffic Counts in Freeway Corridors. Transportation Research B Vol. 28, No. 2. Pp. 141-160.

Chen, Anthony (1997) Formulation of the Dynamic Traffic Assignment Problem with an Analytically Embedded Traffic Model. Doctoral Dissertation, University of California, Irvine.

Cohen, Michael I. (1992) Analysis of the Potential Benefits of In-vehicle Navigation Systems to Alleviate Special-Event Congestion. Master of Science dissertation, University of California, Irvine.

Cremer, M. and Keller, H. (1987) A New Class of Dynamic Methods for the Identification of Origin-Destination Flows. Transportation Research B. Vol. 21B. No. 2.

Jackson, T. (1992) Neural Computing: An Introduction. Institute of Physics Publishing, Bristol and Philadelphia, pp. 1-105.

Jayakrishnan, R. (1992) In-Vehicle Information Systems for Network Traffic Control: A Simulation Framework to study Alternative Guidance Strategies. Doctoral Dissertation, The University of Texas at Austin.

Jayakrishnan et al. (1994) An Evaluation Tool for Advanced Traffic Information and Management Systems in Urban Networks. Transportation Research C, Vol. 2, No. 3, pp. 129-147.

Jayakrishnan, R., Tsai, W.K., and Chen, A. (1995) A Dynamic Traffic Assignment Model with Traffic Flow Relationships. Transportation Research C. Vol. 3, No.1.

Kreyszig, Erwin (1996) Advanced Engineering Mathematics. New Age International (P) Limited Publishers, New Delhi, pp. 280-362.

Ljung, Lennart (1987) System Identification: Theory for the User. PTR Prentice Hall, New Jersey.

Madanat, S.M., et al. (1996) Dynamic Estimation and Prediction of Freeway O-D Matrices with Route Switching Considerations and Time-Dependent Model Parameters. Transportation Research record 1537, TRB, National Research Council, Washington D.C., pp. 98-105.

Nihan, N.L., and Davis, G.A. (1987) Recursive Estimation of Origin-Destination Matrices from Input/Output Counts. Transportation Research B, Vol. 21, No. 2, pp. 149-163.

Okutani, I. (1987) The Kalman filtering approaches in some transportation and traffic problems. 10th International Symposium on Transportation and Traffic Theory. Massachusetts Institute of Technology. Transportation and traffic theory. Elsevier. New York.

Orbix 2: Distributed Object Technology (1997) IONA Technologies PLC, Dublin, Ireland.

Santina, M.S. et al. (1994) Digital Control System Design. Fort Worth: Saunders College Pub. pp. 605-653.

Sherali, H. D. et al. (1994) A Linear Programming Approach for Synthesizing Origin-Destination Trip Tables From Link Traffic Volumes. Transportation Research B. Vol. 28. No. 3. Pp. 213-233.

Sun, Carlos (1998) Use of Vehicle Signature Analysis and Lexicographic Optimization for Vehicle Re-identification on Freeways. Doctoral Dissertation, University of California, Irvine.

Van Aerde, M., et al. (1993) QUEENSOD: A Method for Estimating Time Varying Origin-Destination Demands for Freeway Corridors/Networks. Annual Transportation Research Board Meeting. January. Washington, D.C.

Van Der Zijpp, N., and Hamerslag, Rudy. (1994) Improved Kalman Filtering Approach for Estimating origin-Destination Matrices for Freeway Corridors. Transportation Research Record 1443, TRB, National Research Council, Washington D.C., pp. 54-63

Van Der Zijpp, N., (1997) Dynamic OD-Matrix Estimation from Traffic Counts and Automated Vehicle Identification Data. Transportation Research Board, Washington D.C., Paper No. 970475.

Wagener, Jerrold L. (1980) FORTRAN 77: Principles of Programming, John Wiley & Sons Inc., New York.

Wu, J., and Chang, Gang-Len. (1996) Estimation of Time-Varying Origin-Destination Distributions with Dynamic Screenline Flows. Transportation Research B, Vol. 30, No. 4, Elsevier Science Ltd., pp. 277-290.