Estimation of Origin-Destination Flows for Dynamic Traffic Assignment

by

JOSEF JOSHUA BRANDRISS

B.S. Electrical Engineering (1999)
The Johns Hopkins University

Submitted to the
Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2001

Author .. ..
Department of Electrical Engineering and Computer Science
May 11, 2001

Certified by... .....
Moshe E. Ben-Akiva
Edmund K. Turner Professor of Civil and Environmental Engineering
Thesis Supervisor

Certified by.. .....
Haris N. Koutsopoulos
Volpe National Transportation System Center
Thesis Supervisor

Accepted by .....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Estimation of Origin-Destination Flows for Dynamic Traffic Assignment

by

Josef Joshua Brandriss

Submitted to the Department of Electrical Engineering and Computer Science
on May 11, 2001, in partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering

Abstract

This thesis proposes a framework and method for dynamic origin-destination demand estimation. OD estimation is a critical component of a Dynamic Traffic Assignment system in that it determines the frequencies of drivers' trips through a network.

The OD estimation method presented here allows for tunable optimization to three classes of objectives: Assigned traffic flows, deviation from historical data, and relative proportions in historical data. The method can be easily extended to make use of other sources of information such as direct measurements of OD flows from probe vehicles. The framework is extended to allow for nonnegativity and capacity constraints on the OD flows.

As OD estimation is intended for use in a real-time setting, computational issues are critical, and several simplifications to increase computational efficiency are propsed and evaluted, called the Exact-Match estimator and the Large-Flow estimator.

The algorithms presented are implemented as part of the DynaMIT (Dynamic Network Assignment for the Management of Information to Travelers) traffic estimation and prediction software, which incorporates models for driver route choice and traffic movement simulation.

Thesis Supervisor: Moshe E. Ben-Akiva
Title: Edmund K. Turner Professor of Civil and Environmental Engineering

Thesis Supervisor: Haris N. Koutsopoulos
Title: Volpe National Transportation System Center

Acknowledgments

I would like to use this space to express my gratitude to all of those people who have helped me in so many ways during my time here at MIT. Firstly, I must thank my research advisor Professor Moshe Ben-Akiva for his guidance and assistance. I especially appreciate the time and effort that my thesis supervisor Professor Haris Koutsopoulos has dedicated to me and the other students in the ITS office. He has always been available to talk about ideas and give feedback, and always has time for a friendly chat.

Thank you to all of the researchers at the ITS office for your friendship and advice, and for making our lab an enjoyable and welcoming place to work. To all of you, Marge, Deepak, Angus, Kunal, Dan, Srini, Zhili, and תומר, it has been very nice to know you. I would like to give special thanks to Rama and Manish for all of your help with my work and for your friendship, and to Bruno for getting me started here at the lab.

My research has been supported by grants from the Federal Highway Administration and Oak Ridge National Laboratory, and for this opportunity I am most appreciative.

I cannot forget to mention the people involved with 6.111, Digital Systems Laboratory, the class for which I was a teaching assistant during my first year at MIT. Thank you to Dave Rowe, Jessica Forbess, Ali Tariq, Eladio Arvelo, Jason "Stimpe" Timpe, and Jeff Brown, my fellow TAs, for a most enjoyable experience in the electronics lab. Thanks also to Professors Jim Kirtley and Donald Troxel for providing me with the opportunity to teach and meet so many interesting students, which has been a most rewarding experience.

To all of my friends at MIT and Harvard Hillel, thank you for your companionship and camaraderie. I acknowledge my wonderful roommates of the past years, Victor, Zindel, Shlomo and Barry, for many pleasant times together.

I give special thanks to the Pick family of Newton Centre, Mass., for their extremely generous hospitality these past two years. Your הכנסת אורחים is unparalleled, and I will always appreciate your being my home-away-from-home for so many שבתות and ימים טובים.

And finally, thanks so much to my parents for all your support during these sometimes-difficult two years at MIT.

Josef Brandriss

יוסף יהושע ברנדריס

ל"ג בעומר, תשס"א

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As the nation's highways become more congested for larger and larger parts of the day, much research is being done in the area of Intelligent Transportation Systems (ITS) and Dynamic Traffic Assignment (DTA). It has become clear that the old solutions of building more highways and widening existing ones are too costly or physically infeasible and cannot keep up with increasing demand. ITS aims to alleviate traffic congestion by making more efficient use of the existing infrastructure, and providing tools for planning for future road construction.

Two important ITS elements are advanced traveler information systems (ATIS) and advanced traffic management systems (ATMS). ATIS systems assist travelers with pre-trip planning and enroute decision making. ATMS systems are traffic control systems which can adapt to changing traffic conditions in real-time.

DTA systems, also known as traffic estimation and prediction systems (TrEPS), use advanced traffic models along with surveillance and historical data to estimate and predict conditions in a road network. Output from TrEPS can then be sent to ATIS and ATMS to provide better traffic information and generate better traffic management strategies.

1.1 DynaMIT Overview

Researchers at the MIT ITS Laboratory, under the direction of Professor Moshe Ben-Akiva, have undertaken the development of DynaMIT (Dynamic Network Assignment for the Management of Information to Travelers). DynaMIT is a simulation-based DTA software system which is meant to be installed in traffic management centers (TMCs). DynaMIT generates guidance information based on predicted traffic conditions. Guidance can be either prescriptive or descriptive. Descriptive guidance is the easier to provide; it is simply information on the current state of traffic, such as expected travel times and road link densities. Prescriptive guidance consists of actual recommendations to travelers, such as which route to take in avoiding a congested area. With reliable information as to the current and future states of a road network, drivers can plan their trips better and choose to take an alternate route or decide to make the trip at another time.

To be able to deliver guidance, DynaMIT must first be able to *estimate* the state of the road network,[1] and then *predict* the way the traffic situation will unfold. Typically, we are interested in traffic conditions on the order of an hour or two into the future. Users of DynaMIT-produced guidance will then know about a developing traffic jam and be able to plan accordingly.

Figure 1-1 illustrates the overall structure of the system. This diagram shows the interaction of DynaMIT with a traffic network. Figure 1-2 diagrams the state estimation function of DynaMIT in more detail.

The state of a road network is never known in complete detail and with complete accuracy. Rather, *sensors* will have been deployed at locations throughout the network which collect data on the traffic they observe. Since there are normally many

[1] In this thesis, 'network' refers to the subset of roads and highways under consideration.

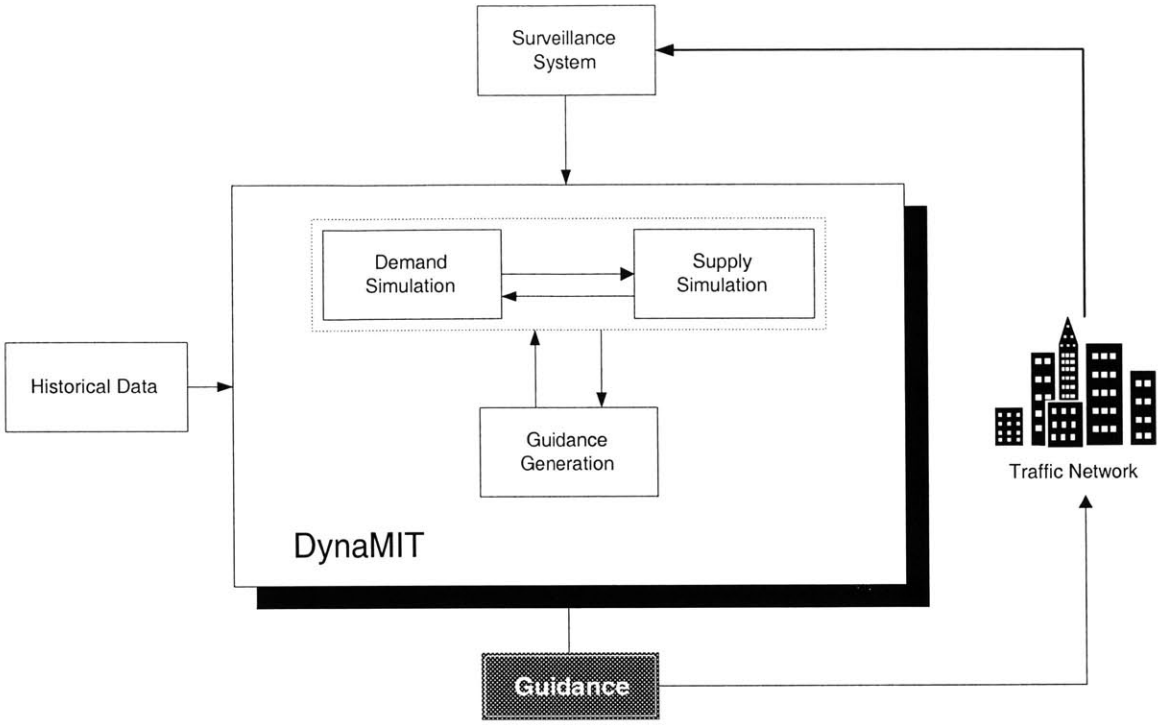

Figure 1-1: The DynaMIT System.

fewer sensors installed than road links, state estimation is required to provide a more complete picture of the traffic situation.

Sensors can be of several types, including optical, camera-based sensors and inductive loop detectors, which consist of a wire loop embedded in the roadway. Sensors will usually provide information such as vehicle counts and speeds and the percentage of that time during which the sensor was occupied by a vehicle.

Clearly, the more sensors there are in a network, the better, but there are practical limits to the number of sensors that can be installed. Furthermore, the most common type of sensors, loop detectors, have reliability issues.

The key modules in DynaMIT used in estimating the state of the network are the *supply simulator* and the *demand simulator*. The supply simulator simulates the

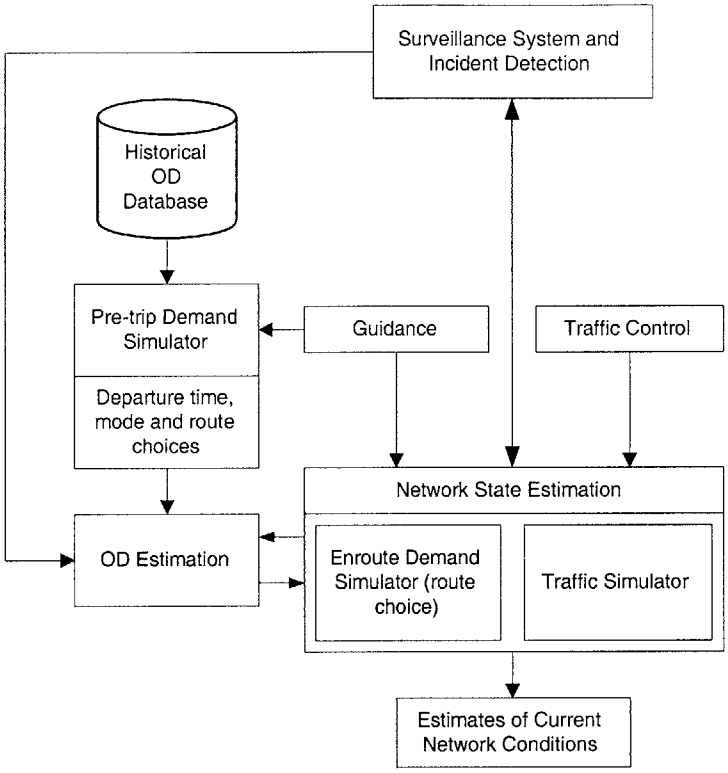

Figure 1-2: State Estimation.

actual movement of vehicles on the network, and the demand simulator simulates the number of vehicles desiring to use the network and the choices of their drivers as to route and departure time. These modules are used iteratively to further refine the estimate of the network state. The supply simulator in DynaMIT is mesoscopic traffic simulation model which uses speed-density relationships and queuing models to capture traffic dynamics. (See Ben-Akiva *et al*, 2001.)

The demand simulator incorporates information about historical flows and predictions about driver choices into the profile it provides to the supply simulator. There is also a distinction in the demand simulator between aggregate and disaggregate

models. The OD matrix itself in an example of aggregate data, since it expresses traffic demand in the aggregate, as flows per unit time. Microscopic, or disaggregate models are used when simulating individual driver decisions – each driver's choices as to departure time and path are simulated individually. DynaMIT makes use of both types of models since historical information is generally available in the form of OD matrices, while the supply simulator requires information about individual drivers.

Figure 1-2 shows the interactions among the various components of DynaMIT's state estimation. The objective of state estimation is to estimate the current state of the network expressed in terms of flows, queues and travel times on the network, and to provide an estimate of the demand on the network. This data is then used in the second major part of DynaMIT, the prediction-based guidance generation.

Prediction-based guidance generation uses the current state of the network and the just-estimated OD vectors to predict future conditions of the network. DynaMIT provides predictions in the form of flows, queues and travel times, which can be used to provide any type of guidance required by the network for the ATIS that might be in place.

1.2 Thesis Focus

The *OD Estimation* component is the main focus of this thesis. Basically, the problem of OD estimation is to use available data, in the form of historical information and measurements of traffic, to estimate the numbers of travelers per unit time from each origin to each destination in the network. While many methods for OD estimation have been developed, few have been implemented as a working component in large-scale, real-time dynamic traffic assignment system. Therefore, in addition to the usual issues of the accuracy of the estimation, problems such as computational efficiency

have to be explored.

Typically, OD matrices were generated using data collected in surveys of drivers. Unfortunately, these surveys are expensive to do and so they take place infrequently. Therefore, the data is usually too outdated for use in real-time applications. In addition, a survey might only represent a "typical" day, but cannot take into account changes in demand such as those resulting from temporary changes in the network, including closing of roads, unusual weather conditions, or special events. So, while historical OD matrices can be used as a starting point, it is beneficial to refine them in real time as current network data becomes available.

This thesis presents a model for estimating OD flows using traffic counts and historical OD data. It is also shown how the model can be extended to use other types of information such as probe vehicles and relative proportions of OD flows to produce more accurate estimates. A constrained version of the OD estimation model is proposed which provides more accurate estimates. Several simplifications are introduced to reduce the computational complexity of the OD estimation problem. It is shown that the speed-up gained by using the simplified models can justify the less-accurate estimates they necessarily will produce.

1.3 Notation

We have attempted to use a uniform notation so that it is easily apparent to what type of data a variable refers. A vector will be indicated by a lowercase letter, such as x_h or y_h. A capital letter, such as A or B is a matrix. A calligraphic letter, such as $\mathcal{X}$ or $\mathcal{Y}$ indicates an augmented vector formed by concatenating individual vectors, and a boldfaced capital letter, such as $\mathbf{A}$ or $\mathbf{B}$ indicates an augmented matrix. A hat above a vector variable, such as $\hat{x}$, refers to an estimate. Normally, in this thesis, all

vectors will be subscripted to indicate the time period they belong to. Occasionally, an unsubscripted vector will refer to a suite of time-period-related vectors, but this will be made clear from the context.

When we wish to indicate a single entry in a vector or matrix, we use parentheses, such as $x_h(j)$ to indicate the jth entry of x_h and $A_h(i,j)$ to indicate the (i,j)th entry of A_h.

Chapter 2

Background and Literature Review

The purpose of this chapter is to formally present the OD estimation problem and related issues in the context of this research, and to review previously developed approaches to the problem. A most comprehensive review can be found in Ashok (1996).

2.1 Estimation Theory

This section gives an introduction to the basic concepts of estimation theory that are used in DynaMIT.

OD estimation relies on results from estimation theory,[1] the body of research involved in the estimation of the values of a group of parameters. The first tasks in estimation are choosing a model for the data and choosing an estimator. For our purposes, we will only choose the linear model for the data; that is, all measurements are linearly related to the data. All of the easiest (that is, closed-form) methods for estimation arise when the linear model is used. The key equation, which we will see

[1]Much of this section is based on material from Kay (1993) and Willsky *et al* (1999).

again and again, is

$$y = Ax + v.$$

$y \in \mathbb{R}^N$ is a vector of observable measurements, $x \in \mathbb{R}^p$ is the parameter to be estimated, and $A \in \mathbb{R}^{N \times p}$ is called the observation matrix, which maps x to y and represents the system whose parameters we are trying to estimate. v is a noise vector with zero mean and covariance C.

The approach we use is to find the estimator which minimizes the error between the actual observation and the observation that would result if our estimate is applied. The error should be minimized in the *least squares* sense. That is, we would like to find the x which minimizes

$$J(x) = \sum_{n=0}^{N-1} (y(n) - a_n' x)^2 = ||y - Ax||_2^2 = (y - Ax)'(y - Ax)$$

where a_n' is the nth row of A. N is the number of data points to be estimated. The least squares estimator does not make any assumptions about the probability distribution of the noise v, just that we know its first- and second-order statistics.

The least squares minimizer can be found in closed form. If we assume that the noise is white with each component of the noise vector uncorrelated with the other; that is, it has covariance matrix $C = \mathbf{I}$, the derivation proceeds as follows. First, we expand the function $J(x)$:

$$\begin{aligned} J(x) &= y'y - x'A'y - y'Ax + x'A'Ax \\ &= y'y - 2x'A'y + x'A'Ax \end{aligned}$$

since the middle terms are scalars. The gradient of this is

$$\frac{\partial J(x)}{\partial x} = -2A'y + 2A'Ax.$$

Setting this to zero gives the ordinary least squares estimate (OLS):

$$\hat{x} = (A'A)^{-1}A'y$$

Of course, this assumes that $A'A$ is of full rank, which will come into play in later discussion. In addition we assume that for $A \in \mathbb{R}^{N \times p}$ that $N > p$. If not, the problem is *underdetermined* and the model parameters cannot be uniquely identified. To see this, consider

$$y = Ax$$

where $A \in \mathbb{R}^{2 \times 3}$ and $x \in \mathbb{R}^3$. Then y can be seen as a projection of x, a vector in three dimensions, onto a two dimensional subspace, and many vectors x can produce the same y. (Please see figure 2-1 for an illustration). In our context, the solution produced from an underdetermined system will not tell us anything about the true situation.

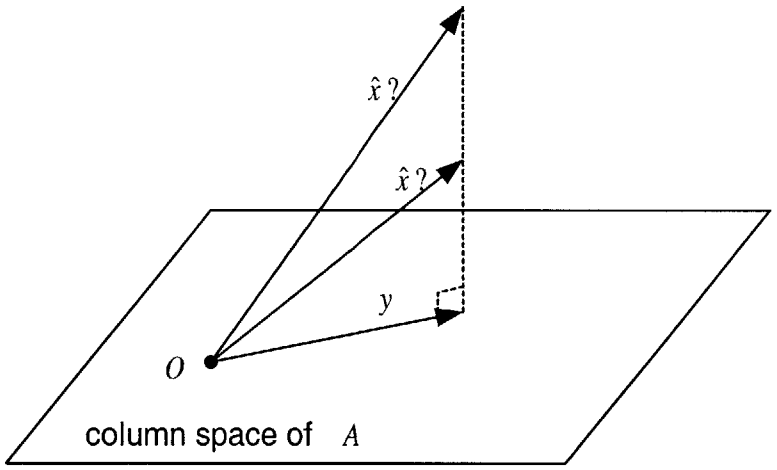

Figure 2-1: Least squares: The underdetermined case.

What about the overdetermined case, where $N > p$? If we were trying to solve the system exactly, we would in general have no hope. However, when we are just trying to find the optimal x in the least squares sense, we need only find the best x such

that $y \approx Ax$. The geometrical situation can be seen in figure 2-2. Here, the situation is reversed. The observed measurements y can be anywhere in y's N-dimensional space. The estimate $\hat{x}$ is chosen so that the measurements $\hat{y} = A\hat{x}$ (that would result if $\hat{x}$ were the true input to the system, and which must be in the column space of A) are as close as possible to y. Note that the error in the least squares estimate's measurement (not to be confused with the estimation error $\hat{x} - x$) is the minimum over all $A\hat{x}$, which must lie in A's column space. Note also that this minimum error is a vector which is orthogonal to the column space of A. The other vector shown, which is the measurement resulting from a suboptimal estimate, has error which is not orthogonal to A's column space.

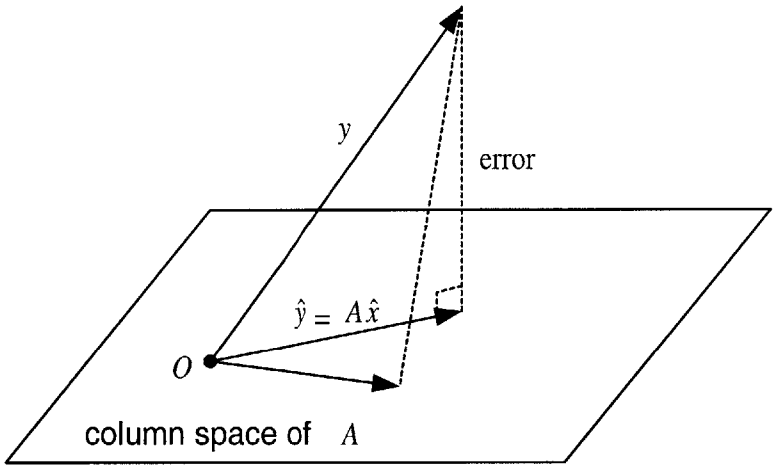

Figure 2-2: Least squares: The standard overdetermined case.

The preceding derivation of the OLS estimator made no assumptions as to the distribution of the measurement errors (noise) v. However, we often do have additional statistical information about the errors in the form of the noise covariance matrix C mentioned above. If this additional source of information is available, and

not exploited, the estimates produced will be inefficient; that is, they do not make efficient use of all available data. This motivates the generalized least squares(GLS) estimator. GLS allows for the incorporation of information about the reliability of measurements in the form of a weighting or covariance matrix. Simple scalar weights can also be used if control over the different parts of the objective function is desired. In addition, if statistical analysis has been performed on the measurement error, a full covariance matrix can be calibrated. For GLS, the objective becomes

$$J(x) = ||B^{-1}(y - Ax)||_2^2 = (y - Ax)'W(y - Ax),$$

where W is an $N \times N$ positive definite weighting matrix. Since W is positive definite, it has a Cholesky factorization $W = BB'$. $C = W^{-1}$ is then the covariance matrix for the measurement noise. The factor B acts as a filter to whiten the noise v. The covariance of the noise transformed by B, or Bv, is then

$$\begin{aligned} E[(Bv)(Bv)'] &= BE[vv']B' \\ &= BCB' \\ &= B(B^{-1}B'^{-1})B' \\ &= \mathbf{I} \end{aligned}$$

Realizing this, the model $y = Ax + v$ can be transformed by multiplying both sides by B:

$$By = BAx + Bv$$

where the noise Bv is now white, so the OLS estimator can be used. So we have

$$\begin{aligned} \hat{x} &= [(BA)'BA]^{-1}(BA)'By \\ &= (A'B'BA)^{-1}A'B'By \\ &= (A'C^{-1}A)^{-1}A'C^{-1}y \end{aligned}$$

Then the GLS estimator is

$$\hat{x} = (A'WA)^{-1}A'Wy. \tag{2.1}$$

A further extension of the least squares (LS) problem involves the addition of side constraints, in the form of inequalities of linear combinations of the estimate. These constraints express prior knowledge we have as to the allowable range of x. The LS problem with linear inequality constraints (LSI) can be expressed as follows:

$$\min_x ||Ax - y|| \quad \text{subject to} \quad Gx \geq h$$

An important special case is nonnegative least squares (NNLS):

$$\min_x ||Ax - y|| \quad \text{subject to} \quad x \geq 0. \tag{2.2}$$

A closed-form estimator is unfortunately not available for these problems. There have been a number of algorithms proposed, however. Lawson and Hanson (1974) give an algorithm for NNLS, and show how LSI can be transformed and solved using an algorithm for least distance programming (LDP), which makes use of the NNLS algorithm. The LDP problem is to minimize $||x||$ subject to linear inequality constraints.

Other algorithms are presented in Stoer (1971), Liew (1976), Escobar and Skarpness (1984), Werner (1990) and Bierlaire *et al* (1991).

2.2 Direct and Indirect Measurements

Ben-Akiva (1987) presents a framework for organizing the types of data available for OD estimation. He refers to *direct measurements*, which is information about the OD flows themselves which can be used directly as a preliminary estimate, and *indirect measurements*, which are data that are used to infer the OD flows.

A direct measurement can be expressed as

$$\hat{x}_h = x_h + u_h \tag{2.3}$$

where $\hat{x}_h$ is the estimate of x_h (OD flows for the time period h) and u_h is a vector of random errors. As described in Ashok (1996), there are several ways to use these direct measurements to generate the estimate $\hat{x}_h$. The two major sources of direct measurements are historical OD tables and information from probe vehicles.

Indirect measurements come from information that is readily observable, but has come from a mapping of the actual OD flows to a range space. The most common, readily available type of indirect measurements are the link counts from roadway sensors. Conveniently, there is a linear mapping from the OD flows to the link counts, which is expressed as follows:

$$y_h(l) = \sum_{p=h'}^{h} \sum_{r=1}^{n_{OD}} A_h^p(r,l) x_p(r) + v_h(l)$$

where v_h is the vector of measurement errors from the link sensors and A_h^p is the *assignment matrix*, which maps the OD flows from time period p to the link counts of time period h. $A_h^p(r,l)$ is the fraction of travelers from the rth OD flow which are detected by sensor l reported during the period h. (More on the assignment matrix later.) $h - h' + 1$ is the maximum number of time intervals needed to travel between any origin and destination on the network. We must sum over all of these time periods since OD vectors from the past contribute to the current state of the network. Of course, if other types of indirect measurements are used, the format of the assignment matrix will change.

The assignment matrix itself is difficult to obtain since it depends on the route choices of the drivers and the traffic conditions in the network, both of which themselves must be estimated. Ashok and Ben-Akiva (2001) present a comprehensive

model for estimation of the assignment matrices while accounting for errors in travel time and route choice models. The authors present models for both offline and real-time estimation. The offline models presented there are more robust, but are impractical for real-time applications.

In DynaMIT the assignment matrix is estimated using the pre-trip demand module and the supply simulator: The historical OD tables are used to generate an initial population of drivers, whose paths are determined using the route choice models. (See Antoniou, 1997.) Then, the supply simulator moves these drivers through the network. The purpose of using the supply simulator is to be able to provide the set of assignment matrices that is needed; for each measurement period h, we require assignment matrices starting at the period h'. This method is the least computationally complicated method available, but suffers serious drawbacks in that it does not take into account the statistical properties of the assignment matrix that can be investigated using Ashok and Ben-Akiva's offline model. Cascetta *et al* (1993, 2001) discuss an iterative method of estimating the assignment matrix, which is discussed in more detail in the next chapter.

The use of other types of direct and indirect measurements have been explored in the literature. Van der Zijpp (1997) uses additional measurements such as OD data from automatic vehicle identification (AVI) which uses image processing technology to recognize license plate numbers at fixed locations in the network to track vehicles' paths. Bottom (1999) and Niver *et al* (2000) propose the use of electronic toll collection (ETC) equipment to record OD data and travel times and speeds. Sun (1999) discusses the use of traffic density rather than traffic flow (that is, vehicle counts) as the key indirect measurement for OD estimation. In congested networks, flows can be misleading. Above a critical level of density, flows begin to decrease, and so flow and travel time do not have a monotonic relationship. On the other hand, the use of

density maintains a convex travel time cost function, and can thus be an indication of whether a road link is in the congested or non-congested regime. Densities can be measured using standard loop detectors by calculating the fraction of time that a loop detector is occupied. Sun introduces an OD-link matrix B which relates OD flows to link densities.

2.3 State-Space Model

A useful abstraction for working with dynamic systems is the *state-space* model. (See Stefani, *et al* (1994) for more.) The state-space model describes the behavior of a system using two simple linear equations, the measurement equation and the transition equation. The measurement equation relates the unknown state of the system to observable data, and the transition equation describes the evolution of the system's state over time. A standard form for a state-space model is

$$y_h = A_h x_h + v_h \tag{2.4}$$

$$x_{h+1} = \Phi_h x_h + u_h + w_h \tag{2.5}$$

where y_h is the measured data and x_h is the state of the system. v_h and w_h are vectors of random errors. u is a deterministic input sequence which also contributes to the evolution of the system state. A_h maps the data to the measurements; it is the observation matrix. Φ_h is the transition matrix, which encapsulates the dynamic behavior of the system. In other words, it describes how the previous states of the system have bearing on the current state.

In our situation, a state space model is useful since many methods have been developed for dealing with this generic type of system. One of the most prominent and computationally feasible for real-time applications is the Kalman filter.

2.3.1 Kalman Filtering

Using the state space model above, it is possible to derive an efficient recursive algorithm for calculating the linear LS estimate x_h. This algorithm, the Kalman filter algorithm, is summarized here. The derivation is presented in Appendix A. Some new notation is required; let $\hat{r}_{n|k}$ denote the linear LS estimate of r at time period n based on observations through time period k.

Let the noise w_h be white and zero-mean, with positive semidefinite covariance matrix Q_h. Similarly, let the measurement noise v_h have positive definite covariance matrix C_h. We assume that the initial state x_0 has a known mean and covariance μ_0 and $\Lambda_0 = C_0$. Then, we initialize the estimator using:

$$\begin{aligned} \hat{x}_{0|0} &= \mu_0 \\ \Lambda_{0|0} &= \Lambda_0 \end{aligned}$$

For each iteration of the filter, we have the following four steps:

1. Generate the next estimate and its associated error covariance. These are the predictor equations, using information only from the state transition equation.

$$\begin{aligned} \hat{x}_{h|h-1} &= \Phi_{h-1}\hat{x}_{h-1|h-1} + u_{h-1} \\ \Lambda_{h|h-1} &= \Phi_{h-1}\Lambda_{h-1|h-1}\Phi'_{h-1} + Q_{h-1} \end{aligned}$$

2. Compute the Kalman gain matrix:

$$K_h = \Lambda_{h|h-1}A'_h(A_h\Lambda_{h|h-1}A'_h + C_h)^{-1}$$

3. Generate the filtered estimate and its associated error covariance. These are the corrector equations, using information only from the measurement equation.

$$\begin{aligned} \hat{x}_{h|h} &= \hat{x}_{h|h-1} + K_h(y_h - u_h - A_h\hat{x}_{h|h-1}) \\ \Lambda_{h|h} &= \Lambda_{h|h-1} - K_hA_h\Lambda_{h|h-1} \end{aligned}$$

4. Increment h and go back to step 1.

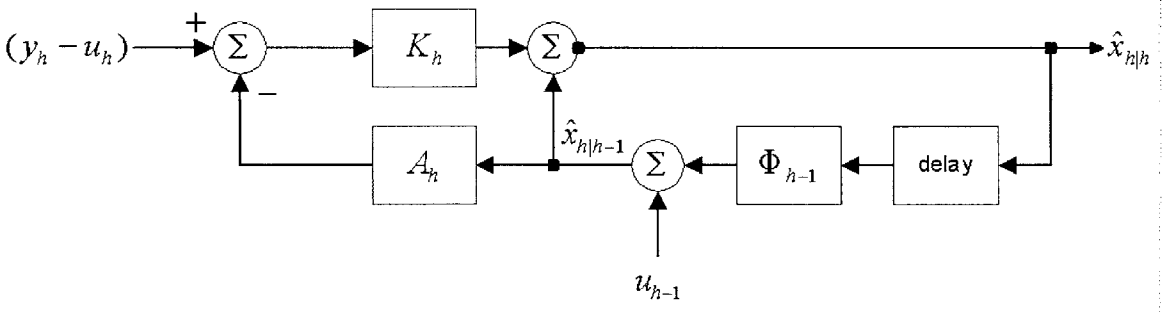

Figure 2-3: The Kalman filter.

There exist a number of variations on the basic Kalman filter algorithm which are worth exploring, one of which bears mentioning here. Chui and Chen (1991) discuss a variant of the Kalman filter algorithm known as the *square-root algorithm*, which is of interest when implementing a numerically robust OD estimation algorithm. The idea involves the Cholesky factorization, which for any positive definite symmetric matrix finds the factorization

$$A = A^c(A^c)'.$$

A^c can be viewed as the generalized square root of the matrix A. In a special case, if

$$A = \text{diag}(a_{11}, \ldots, a_{nn}),$$

then the Cholesky factorization gives

$$A = A^c(A^c)' = \begin{bmatrix} \sqrt{a_{11}} & & \\ & \ddots & \\ & & \sqrt{a_{nn}} \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & & \\ & \ddots & \\ & & \sqrt{a_{nn}} \end{bmatrix}$$

The advantage of going to the square-root is that small numbers become larger and large numbers become smaller, so that number are of more of the same order, and computations can be more accurate.

An advantage of the Kalman filter algorithm is that generating a new estimate does not require any inversions. Each iteration of the algorithm requires 4 matrix-matrix multiplications and 3 matrix-vector multiplications, compared to 2 and 2 for the GLS estimator. Note that the Kalman gain matrix K_h and the error covariances do not depend on the real-time measurements coming in, and can be pre-computed and stored if computational speed is an issue.

The Kalman filter has an additional advantage in that the estimates that it produces incorporate the dynamic behavior of the system as described by equation 2.5. The one-step GLS estimator as described above does not incorporate any system dynamics; rather, it focuses on the measurements obtained during the current time interval and ignores knowledge about any other points in time. A GLS estimator can be derived which takes the entire record of system behavior into account, but this requires the solution of *all* the time intervals simultaneously at each subsequent time interval. Ashok (1996) shows the equivalence of this *batch* GLS estimator with the Kalman filter estimates. We will, however, derive a version of the single-time-interval GLS estimator which does take the system dynamics into account.

Chapter 3

Estimation Model

In this chapter the theoretical basis for the OD estimation model is presented. First, the DynaMIT demand simulation model is presented in greater detail. Then, for OD estimation, we start with the most basic model that can be implemented, one which estimates the OD flows based on link counts from sensors and historical OD information. We then proceed with enhancements to the model based on additional information sources, and simplifications to reduce computing time.

3.1 Demand Simulation in DynaMIT

3.1.1 Flow of Execution

DynaMIT begins a new execution cycle at set intervals. Each execution cycle contains two major steps: state estimation and prediction-based guidance generation. Please see figure 1-2 for a block diagram of state estimation. Please also see figure 3-1 for a block diagram of the demand simulation component of DynaMIT.

To review, the function of state estimation is to provide estimates of the current

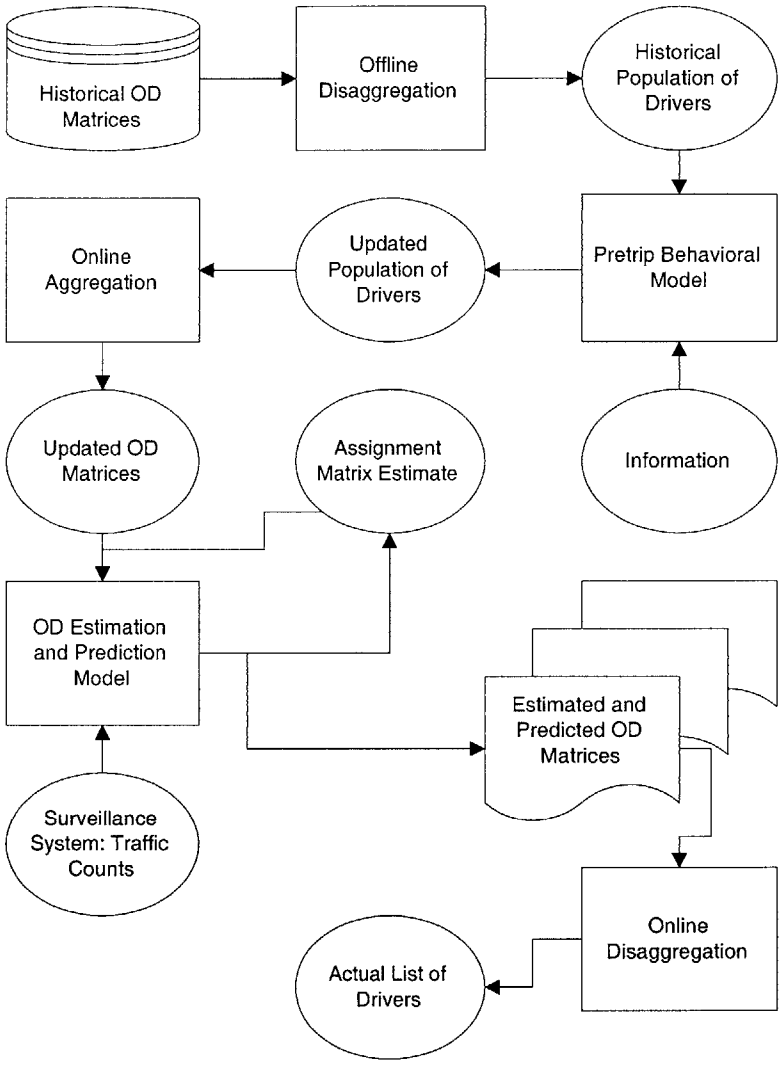

Figure 3-1: Demand Simulation.

network state in terms of OD flows, link flows, queues, densities and travel times. The demand estimation, using inputs from historical databases and network surveillance (sensors, etc.), provides the input to the traffic simulator. The traffic (supply) simulator then provides an estimate of variables such as flows, speeds and queues.

It may be necessary to go through several iterations of state estimation until a consistent estimate is obtained. This is because there are two main sources of uncertainty in the OD estimation module: the OD vector itself, and the assignment matrix. To estimate the assignment matrix it is necessary to have a good idea of the OD vector, and to estimate the OD vector one must have a good assignment matrix. So some feedback is necessary to achieve convergence between the two.

The prediction-based guidance generation uses predicted OD vectors to predict future traffic conditions. There can also be feedback and several iterations in this part of the program, as different guidances are tried until one is selected with the desired effects. After the OD vector is estimated for the current time interval, the prediction process is run to generate predicted OD vectors for the number of time intervals in the prediction horizon, as described in section 3.2.1. The supply simulator is then used to provide predictions of flows, queues and travels times in the network.

It is useful for the purposes of this discussion to give concrete examples for the time intervals used. For reporting sensor counts and estimation OD vectors, it is useful to choose a time interval length that is short enough to capture reasonable temporal variations in the network conditions while not so short as to make computation unwieldy. A reasonable time interval length is 15 minutes.

3.1.2 Assignment Matrix

Typically DynaMIT would be executed whenever there is a need for traffic prediction, such as during rush hours, special events, or whenever an incident is detected on the

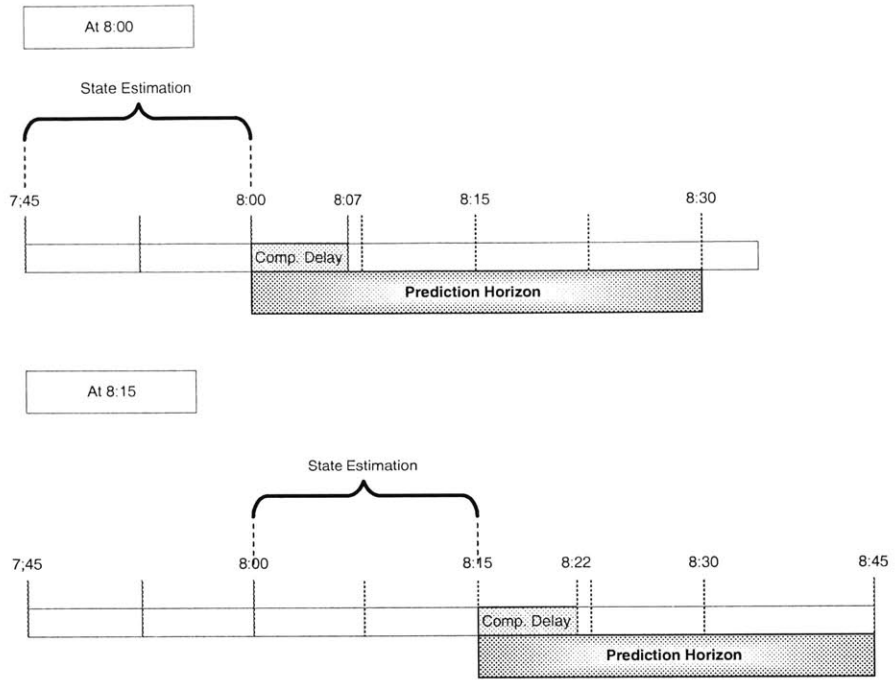

Figure 3-2: The rolling horizon in DynaMIT.

network. For example, for the morning rush hour DynaMIT could be started at 6:15 AM. The state estimation would then perform its estimation for the 15-minute period 6:00 – 6:15. Since this is the first interval of the day for DynaMIT, the only OD data available is from the historical database, which, again, could come from survey data or a database built from previous days' runs. Also, no current assignment matrix is available.[1] Therefore, at least one "dry run" of the supply simulator must be done before OD estimation can be performed.

[1]Ashok (1996) envisions a framework for estimating a stochastic assignment matrix from travel time and route choice models in a closed loop with OD estimation. With speed of computation as a priority, however, we have implemented the following ad hoc approach for estimating the assignment matrix.

The assignment matrix is estimated after each run of the supply simulator. Good results here depend heavily upon the route choice model and the accuracy of the supply simulator. DynaMIT currently generates a set of reasonable paths (which don't contain any cycles). Each driver generated is assigned a path based on calibrated discrete choice (logit) models. Developing new route choice models is an area of ongoing intense research.

DynaMIT keeps track of every driver in the supply simulator, so in DynaMIT itself, the assignment matrix can be computed with complete accuracy. Each driver is tagged with its origin, destination, current location and departure time. Suppose we want to compute the assignment matrix A_h^h of effects of the current OD vector to the current interval. Then we can look at each link in turn and count the number of drivers on each link from each OD pair. For a given link, we then know the proportion of drivers each OD pair contributes to that link.

To illustrate, suppose we have a simple network shown in figure 3-3.[2] This network contains 10 links and 10 nodes. There are then $10 \times (10 - 1) = 90$ possible OD pairs. In most cases, however, not all OD pairs are active. For OD estimation, DynaMIT only considers the OD pairs for which there are entries in the historical database. Even though on any given day trips can be made for an OD pair which had not been considered before, the proliferation of OD pairs would make computation prohibitively slow.

Suppose that in our example that the only loaders (active origins and destinations) are the leaves of the network (that is, at notes 1, 4, 7, 8, 9 and 10). Suppose we have 3 active OD pairs: (1,4), (1,7) and (8,10), each with equal demand, and that every driver traverses the network in a single time interval. For each of the first two OD

[2]This will be referred to as the "Florian Network," and it is an example courtesy of Prof. Mike Florian of the University of Montréal.

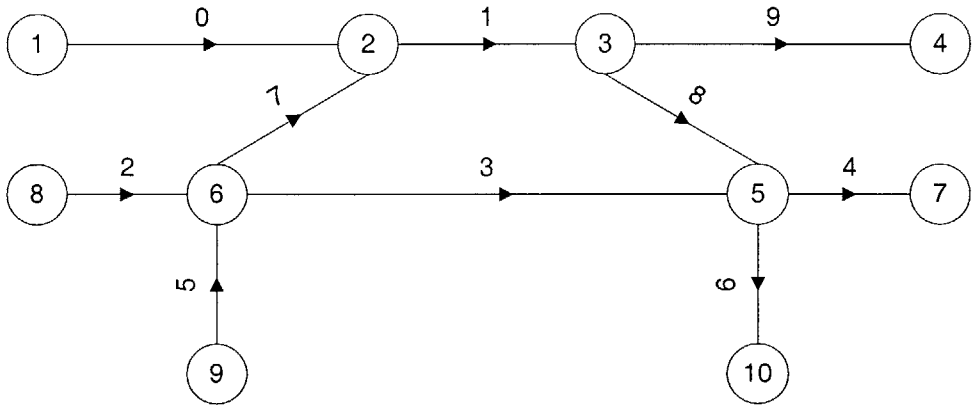

Figure 3-3: A Simple Network.

pairs, only one path is possible, but two paths are possible for OD pair (8,10). Suppose drivers are equally likely to take either path. Consider link 8, which goes from node 3 to node 5. Then row 10 of the assignment matrix will be

$$\begin{bmatrix} 0 & 1 & \frac{1}{2} \end{bmatrix}$$

since no drivers from pair (1,4) take this link, all the drivers from pair (1,7) take the link and half of the drivers from (8,10) take the link. For this example we have assumed that all drivers finish their trips within a single time interval. Actually, of the drivers detected on the link, some may have departed during a previous time interval, but DynaMIT can take this into account as stated previously.

3.1.3 Iterative Estimation of Assignment Matrix and OD Flows

Once an assignment matrix has been generated, OD estimation can begin. By the time DynaMIT executes after 6:15 for the 6:00 – 6:15 interval, we already should have

real-life sensor data from that interval. The OD estimation process, as described later, produces an OD vector which can be fed back into the supply simulator to produce a new estimate of the assignment matrix, and so on. Once a satisfactory OD vector has been generated (if the error between the estimated assigned sensor counts and actual sensor counts is below a certain threshold, or after a maximum number of iterations) the prediction is run for the desired horizon length. If, for example, the horizon length is 30 minutes, the 2 OD vectors for 6:15 – 6:30 and 6:30 – 6:45 would be generated by the OD prediction process, and the supply simulator would be run for 6:15 – 6:45.

The goal of the iteration between the supply simulator and OD estimation is that the assignment matrix estimates and OD vector estimates will both converge to a fixed point, which is the optimum solution when both estimation problems are considered together. As of now, DynaMIT feeds the estimated OD vector back into the supply simulator directly, with no preprocessing of the OD vector. In fact, it is possible to use some intelligent processing to aid in the convergence of this estimation scheme. Cascetta *et al* (2001) suggests some methods, including a smoothing algorithm known as the method of successive averages (MSA), and criteria for the existence of a fixed point.

The following is an intuitive derivation of the smoothing algorithms based on estimation theory. The MSA is based on a simple estimation problem known as "DC Level in White Noise" presented in Kay (1993). It is posited that there exists a certain target true OD vector x_h^*, and the successive OD estimates that are obtained are noisy measurements of the true vector x_h^*, or

$$\hat{x}_h^k = x_h^* + w^k$$

where k is the current iteration of the estimation, and w^k is a vector of white noise,

where w^k is assumed to be uncorrelated with estimates produced at previous iterations, and the individual components of w^k are uncorrelated with each other. We could estimate x_h^* by taking the current "measurement" $\hat{x}_h^k$, like is currently done in DynaMIT, but this a statistically inefficient estimator in that it ignores all of the previous measurements of x_h^*. In fact, for a single component of the vector to be estimated, the variance of the simple estimate is k times that of the MSA estimator.

Suppose we concentrate on just a single element of the target OD vector, where we are trying to estimate a constant level $x_h^*(j)$ in the presence of white noise, so that

$$\hat{x}_h^k(j) = x_h^*(j) + w(j)$$

where $\hat{x}_h^k(j)$ is the current "measurement" (really, the current estimate) of OD flow j and $w(j)$ is white noise with variance σ^2. The simple estimator $\tilde{x}_h^*(j) = \hat{x}_h^k(j)$ has variance σ^2. The MSA estimator involves taking the average of all of the measurements received, or

$$\hat{x}_h^*(j) = \frac{1}{k}\sum_{l=1}^{k} \hat{x}_h^l(j)$$

Its variance is

$$\begin{aligned} \text{var}(\hat{x}_h^{*k}(j)) &= \text{var}\left(\frac{1}{k}\sum_{l=1}^{k} \hat{x}_h^l(j)\right) \\ &= \frac{1}{k^2}\sum_{l=1}^{k} \text{var}(\hat{x}_h^l(j)) \\ &= \frac{\sigma^2}{k} \end{aligned}$$

which is much smaller than the simple estimator's variance, indicating that the MSA provides a better estimate.

Here, we have successive vector observations of a target assumed to be a fixed point. The LS estimate is

$$\hat{x}_h^{*k} = \frac{1}{k}\sum_{l=1}^{k} \hat{x}_h^l$$

To save us from recalculating the average at each step, which would also involve storing the estimates from all previous iterations, we realize that the estimate can be written

$$\begin{aligned} \hat{x}_h^{*k} &= \frac{1}{k}\left(\sum_{l=1}^{k-1} \hat{x}_h^l + \hat{x}_h^k\right) \\ &= \frac{k-1}{k}\hat{x}_h^{*k-1} + \frac{1}{k}\hat{x}_h^k \end{aligned}$$

This method is optimum if it is the case that each estimate of the OD vector (or "measurement" in this context) from the first and on, has the same variance properties (that is, that the noise process w has the same variance at each iteration). We know this is far from the truth, since we expect that the first estimate will be worse than successive ones, suggesting that each successive w^k has smaller variance than the previous. In this case we have a linear model with a noise process whose covariance matrix is not the identity, but $\Sigma_k \mathbf{I}$, where $\Sigma_k = [\sigma_1^2 \ldots \sigma_k^2]$ is generated by some decaying function of l, such as $\sigma_l^2 = ae^{-al}$.[3] In this case, the LS estimate is (see Kay (1993) section 8.7 for a derivation)

$$\hat{x}_h^{*k} = \hat{x}_h^{*k-1} + \frac{1/\sigma_k^2}{\sum_{l=1}^{k} \frac{1}{\sigma_l^2}}(\hat{x}_h^k - \hat{x}_h^{*k-1})$$

This algorithm is similar to what Cascetta *et al* call the method of successive averages with decreasing refreshing (MSADR).

This method could of course also be used to smooth the assignment matrix estimates, but would be less computationally efficient due to the larger size of this matrix. Cascetta *et al* have shown experimentally that the simple algorithm converges faster,

[3]Note the distinction between the covariance matrix of the noise w^k and the noise process for a specific OD pair $w(j)$. Here, we are saying that the covariance matrix of the noise w^k is $\sigma_k^2\mathbf{I}$, and the covariance matrix of the noise w^{k-1} is $\sigma_{k-1}^2\mathbf{I}$. The covariance matrix of the noise *process* $w(j)$ (which has been observed until the current time interval k) is $\Sigma_k\mathbf{I}$.

but the MSA and MSADR have better convergence properties. Cascetta prefers the MSADR because of its more-intelligent smoothing properties.

3.2 Basic Model Formulation

Consider a network with n_{OD} OD pairs and n_l available sensors to provide link counts. Following the notation of Ashok *et al*, we write:

- x_h is the OD vector for time interval h; that is, it represents the number of vehicles departing during time interval h for each OD pair. This is the vector to be estimated.
- x_h^H is the corresponding historical OD vector available.
- $\hat{x}_h$ is the *estimated* OD vector from time interval h.

At the most basic level, the information available to us is in the form of link counts. We express the counts recorded for all of the sensors in the time period h by y_h. As stated before, there is a linear relationship between the OD flows and the sensor counts, expressed by:

$$y_h(l) = \sum_{p=h'}^{h} \sum_{r=1}^{n_{OD}} A_h^p(r,l) x_p(r) + v_h(l)$$

where h' is the earliest departure time interval for which vehicles remain on the network. In matrix form this becomes

$$y_h = \sum_{p=h'}^{h} A_h^p x_p + v_h. \tag{3.1}$$

where $A_h^p \in \mathbb{R}^{n_l \times n_{OD}}$ is the assignment matrix of contributions of x_p to y_h. Each entry $A_h^p(r,l)$ is the ratio of drivers from OD pair r which are detected by sensor l

during time period h to the total demand for OD pair r. The ratio is calculated for each departure time interval p for which vehicles remain on the network, resulting in a suite of $h - h' + 1$ assignment matrices for each time interval.

Here, we clearly have a linear model for the system, so linear least squares estimation can be used. However, as it stands, the system is underdetermined. That is, in general a network will have many fewer sensors than possible OD pairs, and so without further information we have no hope of getting a meaningful estimate of the OD vector.

An easy first step is to include direct measurements in the formulation. One category of direct measurements is historical OD vectors, which can be from the historical database containing survey data and previous estimation results. For the first iteration of the first time interval, of course, the only historical OD vector available is from the historical database. In any case, the objective function now becomes

$$\underset{\hat{x}_{h'},\ldots,\hat{x}_h}{\arg\min} \left(w_1 \left\| y_h - \sum_{p=h'}^{h} A_h^p \hat{x}_p \right\| + w_2 \sum_{p=h'}^{h} \left\| x_p^H - \hat{x}_p \right\| \right) \tag{3.2}$$

We may choose weights w_1 and w_2 to indicate the relative confidence we would like to ascribe to either form of measurement. If more fine control is desired over the weighting, we can, instead of using the scalars w_1 and w_2 use diagonal weighting matrices W_1 and W_2. For example, if we know that a certain sensor is malfunctioning, and we have less confidence in its output, we can assign its entry a lower weight in W_1. After a network is well-calibrated, and a good historical database is available, we would have calibrated covariance matrices for the errors, which should incorporate such information as the presence of a faulty sensor. In that case, the scalar weights can still be used.

If full variance or covariance matrices are available, the objective function becomes

$$\underset{\hat{x}_{h'},\ldots,\hat{x}_h}{\arg\min}\left[\left(y_h-\sum_{p=h'}^{h}A_h^p\hat{x}_p\right)'R_h^{-1}\left(y_h-\sum_{p=h'}^{h}A_h^p\hat{x}_p\right)+\sum_{p=h'}^{h}\left(x_p^H-\hat{x}_p\right)'Q_p^{-1}\left(x_p^H-\hat{x}_p\right)\right]$$

where R_h is the covariance matrix of the indirect measurements, and Q_h is the covariance matrix of the direct measurements.

3.2.1 Prediction and the Idea of Deviations

This section discusses an enhancement to the direct measurement equation just presented which incorporates information about the dynamics of the system. Once the OD vector has been produced for time interval h, DynaMIT must then predict the state of the network for a certain horizon length into the future. This is done using the supply simulator, which requires OD vectors as inputs. In the state-space model, we have proposed that the OD vectors develop through time according to

$$x_{h+1}=\Phi_h x_h+w_h+u_h. \tag{3.3}$$

We additionally refer to this development of the OD flows in time as the *autoregressive process*. The matrix Φ_h expresses the relationship between the OD vectors from time interval h and $h+1$. This matrix must be calibrated off-line using historical data. Note that we have assumed for the purpose of our state-space model that only the immediately previous OD vector is related to the current one through the transition equation. In the simplest form of the transition equation, we assume that effects from other OD vectors are negligible.[4] In the absence of good estimates for the Φ's, we can simply use the latest OD estimate for the prediction intervals, which is equivalent to setting Φ_h to the identity matrix. This can still provide good

[4]Additional research on offline calibration and building the historical database can be found in Balakrishna (2001).

results for the prediction, since the supply simulator for the prediction horizon starts with an already-loaded network; that is, all of the congestion that is known to exist is perpetuated during the prediction interval. However, this does not make use of all of the information available to us, and results in a statistically inefficient estimator.

Ashok and Ben-Akiva propose the idea of *deviations* to mitigate this deficiency of the simple state-space prediction model. As it stands, the objective function of equation 3.2 strives to keep the estimate close to the historical OD flows. However, this obscures more valuable information that is available about the development of OD flows over the course of a day. We might know, in a holistic sense, that rush hour starts at 7:00 AM and continues until 9:00, and that there is an evening rush hour from 4:30 PM until 6:30. People leave for lunch around noon, and may travel to events in the evening. So we can observe trends in travel demand over the course of many days. This knowledge is expressed numerically in the transition equation (equation 3.3) of the state-space model for use in the prediction of the OD vectors.

Since the deviations subsume important information about the system dynamics, they must also be incorporated into the GLS formulation. Rather than use the simple for for direct measurements as expressed in equation 2.3, we can write, in deviations form,

$$x_h - x_h^H = \sum_{p=1}^{h-1} F_h^p (x_p - x_p^H) + w_h \tag{3.4}$$

where $F_h^p \in \mathbb{R}^{n_{OD} \times n_{OD}}$ is the matrix of effects of the previous time intervals' deviations on the current time interval's deviations. Note that now we have enhanced the definition of the F matrices to indicate the effects of any previous time interval on the current time interval.

This new formula gives us freedom from blindly following the historical OD flows in our estimates. For instance, if today is an abnormally high-traffic day, the true OD flows will exceed the historical tables. However, the estimator as expressed earlier

will contradict the knowledge gained from the sensor measurements which indicates the increased demand. The use of deviations allows the estimator to "realize" that previous time intervals have deviated from the historical flows, and that today is not a normal traffic demand day. To make this more concrete, suppose that DynaMIT is started at 6:00 AM on a high-traffic day. The sensors indicate a high demand, but the historical tables pull the estimate down. Suppose we have set the weights on the estimator such that the two conflicting sets of data are met halfway. Then a deviation is recorded at the first time interval, 6:00–6:15. Knowing through the F_h^p's how the deviation is to propagate, the estimate of 6:15–6:30 will be further refined.

Even if we do not have a calibrated historical database of the F_h^p's, the method of deviations can still allow us to refine the estimates. We can, as described above, set the F_h^p's to be diagonal matrices, and still have the objective that tries to agree with the deviations. For the F matrices to be diagonal means that we are assuming that only the temporal relationships between specific OD flows matter, and that contributions from other OD pairs do not matter much. For an ad hoc estimate of the F's, in the absence of a historical database, we could additionally hypothesize that as we look farther back in time, previous intervals' deviations have less and less relevance to the current time interval, and set the scaling of the F matrices accordingly.

Notice that equation 3.4 is expressed in a very general form. We have allowed for each interval of the day to be included in the calculation of the latest estimate. What this is saying is that we believe that the deviations of 6:00–6:15 AM have some relevance to those of, say, 7:00–7:15 PM. As we get later in the day, this means that the right hand side of equation 3.4 will become more and more time-consuming to calculate. And so far we have only assumed that DynaMIT is to be run on a day-by-day basis, but theoretically deviations from 7:00–7:15 PM on the previous day have an effect on the deviations of 6:00–6:15 AM today. For this discussion we define the

degree of the autoregressive process to refer to the number of intervals back into the past we consider. If we set the degree of the process to be 4, we indicate that we only consider deviations of the past hour in our estimation and prediction. The optimal degree of the autoregressive process can also be calibrated off-line.

The degree of the process is important if we want to use the simplification that the F_h^p's be diagonal matrices. Again, this simplification may be needed for two reasons:

- in the absence of a well-calibrated historical database of F matrices, or
- if computational speed is an issue.

In general, the calibrated F matrices will be full, and therefore each estimation performed will require a full matrix-vector multiplication as many times as the degree of autoregressive process used. If computation time is at a premium, it may be desirable to use diagonal F matrices even if full ones can be obtained.

We can now revise the objective function of equation 3.2:

$$\min_{\hat{x}_p} \left(w_1 \left\| y_h - \sum_{p=h'}^{h} A_h^p \hat{x}_p \right\| + w_2 \left\| x_h^H - \hat{x}_h + \sum_{p=\max(h-d_a,1)}^{h-1} F_h^p (x_p^H - \hat{x}_p) \right\| \right) \tag{3.5}$$

where d_a is the degree of the autoregressive process used. As before, we write the weights in the objective function as scalars for simplicity, but full covariance matrices can be used in the estimator. Note that for the direct measurements we have the summation index p start from $\max(h - d_a, 1)$. This is because for the first d_a time intervals there are fewer than d_a previously-estimated OD vectors.

In addition, we rewrite the autoregressive process with its degree:

$$x_h = x_h^H + \sum_{p=h-d_a}^{h-1} F_h^p (x_p - x_p^H) + w_h \tag{3.6}$$

3.2.2 Standard Estimator

Note that in this formulation, each execution of OD estimation requires the estimation of multiple OD vectors. This is because link counts at a specific time interval include not only those vehicles which departed during that interval, but also vehicles which have departed during past intervals that remain on the network and have not yet reached their destinations. Specifically, we must reestimate all of the OD vectors $x_{h'}, \ldots, x_h$. Let us define the vector

$$\mathcal{X}_h = [x_h \quad x_{h-1} \quad \ldots \quad x_{h'}]'$$

This formulation can be expressed as a linear least squares problem of the form $\mathcal{Y} \approx \mathbf{A}\mathcal{X}$ as discussed above. It is important to now view $\mathcal{Y}$ in the canonical form as the vector of *all* measurements available, whether direct or indirect. $\mathcal{X}$ is still the vector to be estimated, but now $\mathbf{A}$ must be constructed properly so that the mapping from $\mathcal{X}$ to $\mathcal{Y}$ is correct.

Specifically, $\mathcal{Y}$ is the vector of sensor counts, concatenated with the historical (or previously estimated) OD vectors from time intervals h' through h. So, we write

$$\mathcal{Y}_h = \begin{bmatrix} & y_h & \\ x_h^H & + & \sum_{p=\max(h-d_a,1)}^{h-1} F_h^p (x_p^H - \hat{x}_p) \\ \hat{x}_{h-1} & + & \sum_{p=\max(h-1-d_a,1)}^{h-2} F_{h-1}^p (x_p^H - \hat{x}_p) \\ & \vdots & \\ \hat{x}_{h'} & + & \sum_{p=\max(h'-d_a,1)}^{h'-1} F_{h'}^p (x_p^H - \hat{x}_p) \end{bmatrix}$$

At the beginning of the day, there will be fewer previous estimates than the degree of the autoregressive process. For those terms, just set $\hat{x}_p = x_p^H$. For example, for the first time interval of the day, we have no previous estimates and therefore we have no choice but to use the objective that the deviation between the current estimate and the historical vector approach 0.

$\mathbf{A}_h$ is constructed like so:

$$\mathbf{A}_h = \begin{bmatrix} A_h^h & \dots & A_{h'}^h \\ & \mathbf{I}_{(pn_{OD})} & \end{bmatrix} \in \mathbb{R}^{n_l + pn_{OD} \times pn_{OD}} \tag{3.7}$$

For convenience, define $p = h - h' + 1$ as the number of time intervals which have an effect on the current state. To check the dimensions, note that $\mathcal{Y}_h \in \mathbb{R}^{pn_{OD}+n_l}$ and $\mathcal{X}_h \in \mathbb{R}^{pn_{OD}}$. The upper section of $\mathbf{A}_h$ is of dimension $n_l \times pn_{OD}$, while the lower section an identity matrix of dimension pn_{OD}.

We must consider the computational feasibility of this problem. In the GLS method, (see equation 2.1) each estimation involves inverting the matrix $A'WA$, where $W = C^{-1}$, and C is the error covariance matrix. It is possible that the assignment matrix by itself is too sparse for the matrix $A'WA$ to be invertible. Consider, for example, a test case one wants to run, in which few of the possible OD pairs in the network are active. Many links on the network may not be used at all in this test case, leaving rows of zeros in the assignment matrix. Suppose that W is a diagonal weighting matrix. Then, even if there is a single row of zeros in A, the product $A'WA$ will be badly conditioned, close to singular, and therefore not invertible. This can be seen most easily through an example. Suppose A is a 5×6 full-rank matrix, and then the third row is set to all zeros. Then A' is a 6×5 matrix with the third column all zeros. Both A' and A then have rank 4, and the product $A'WA$ has rank no greater than 4, and is therefore not invertible.[5]

However, augmenting the problem with the direct measurements will always have the added bonus of resolving this difficulty. The matrix $\mathbf{A}_h$ as defined in equation 3.7 clearly has rank pn_{OD} because of the presence of the identity matrix in its lower sector. Therefore, the product $\mathbf{A}_h'\mathbf{A}_h$, which is a $pn_{OD} \times pn_{OD}$ matrix, will be of full rank and consequently invertible.

[5]since we know concerning the rank of a product that $r(AB) \leq r(A)$ and $r(AB) \leq r(B)$.

3.3 Key Simplifications

At this point, we have enough information to estimate the p latest OD vectors using a GLS estimator or the Kalman filter algorithm. However, it is important to consider the computational cost involved in the estimation as formulated. Since flows from p time intervals contribute to the measurements (sensor counts) of the current time interval, we simultaneously estimate all OD vectors for these intervals. The key computations in the GLS method involve the inversion of the matrix $A'WA$ and matrix-matrix multiplications, both of which have complexity $O(n^3)$, where n is the size of the matrix. As the network gets larger, computation time increases dramatically because of the size of the matrix, but there is another factor as well: the number p can grow larger since it takes vehicles more time to traverse the network. In addition, if the network becomes more congested, the number p will also increase. Therefore, we use an approximation which is well-suited to the design of DynaMIT.

Basically, we would like to take the previous OD vectors out of the estimation formulation. Ashok (1996) has demonstrated that this approximation does not add significantly much to the estimation error.

We can rewrite the measurement equation (3.1) as follows:

$$y_h = A_h^h x_h + \sum_{p=h'}^{h-1} A_h^p \hat{x}_p + v_h$$

The rightmost term should now be treated as a constant; given the previously estimated OD vectors and assignment matrices, we can subtract the vehicles from those periods which remain on the network.

In DynaMIT, as it turns out, this is quite straightforward to implement, not even requiring any matrix-vector multiplications. As DynaMIT runs, all of the estimated OD vectors are fed into the supply simulator. Each vehicle on the network has identifying information, including its departure time, origin and destination. So,

directly from the supply simulator's list of drivers, those drivers which have departed in previous time intervals can be tallied by OD pair. These totals are then subtracted from the sensor counts reported by the surveillance system. We indicate these totals as p_h, the vector, by link, of vehicles remaining on the network in time interval h which have departed prior to h. Now, the measurement equation becomes

$$y_h - p_h = A_h^h x_h + v_h \tag{3.8}$$

and we would like to find

$$\arg\min_{\hat{x}_h} \left(w_1 \left|\left|(y_h - p_h) - A_h^h \hat{x}_h\right|\right| + w_2 \left|\left| x_h^H - \hat{x}_h + \sum_{p=\max(h-d_a,1)}^{h-1} F_h^p (x_p^H - \hat{x}_p) \right|\right| \right).$$

(Again, w_1 and w_2 are used for notational convenience. The above equation can be written in a dot-product formulation using full covariance matrices.) Note that in this case, though we use deviations from previous time intervals in the objective function, the previous estimates $\hat{x}_p$ are now considered to be constants. We can now redefine the augmented vectors and matrix used in the least squares estimator:

$$\mathcal{X}_h = x_h \tag{3.9}$$

$$\mathcal{Y}_h = \begin{bmatrix} (y_h - p_h) \\ x_h^H + \sum_{p=\max(h-d_a,1)}^{h-1} F_h^p (x_p^H - \hat{x}_p) \end{bmatrix} \tag{3.10}$$

$$\mathbf{A}_h = \begin{bmatrix} A_h^h \\ \mathbf{I}_{n_{OD}} \end{bmatrix} \tag{3.11}$$

The estimator is then

$$\hat{x}_h = (\mathbf{A}_h' \mathbf{C}_h^{-1} \mathbf{A}_h)^{-1} \mathbf{A}_h' \mathbf{C}_h^{-1} \mathcal{Y}_h \tag{3.12}$$

where

$$\mathbf{C}_h = \begin{bmatrix} \frac{1}{w_1} R_h & \mathbf{0}_{(n_l \times n_{OD})} \\ \mathbf{0}_{(n_{OD} \times n_l)} & \frac{1}{w_2} Q_h \end{bmatrix}$$

The covariance matrix $\mathbf{C}_h$ should be block diagonal, indicating that the indirect measurements and direct measurements are uncorrelated. The block components R_h and Q_h are the covariance matrices associated with the indirect measurements and direct measurements, respectively. These matrices are calibrated off-line and become part of a historical database. The weights w_1 and w_2 indicate the level of confidence in the sensor counts and assignment matrix as opposed to the historical database and deviations. Again, if the historical database has not yet been developed, or more computational efficiency is desired, then the matrices R_h and Q_h can be set to the identity matrix, causing $\mathbf{C}_h^{-1}$ to be simply a diagonal weighting matrix. For a diagonal $\mathbf{C}_h$, instead of 4 matrix-matrix multiplies, the estimator requires only 2.

The prediction is run according to the autoregressive process. We use the notation $\check{x}_g$ to indicate a predicted OD vector. For each interval g for which a prediction is desired, we have

$$\check{x}_g = x_g^H + \sum_{p=\max(g-d_a,1)}^{h} F_g^p(\hat{x}_p - x_p^H) + \sum_{p=h+1}^{g-1} F_g^p(\check{x}_p - x_p^H)$$

This formula is broken into two sums to indicate that we may be predicting several intervals beyond the last estimate, so that the deviations can only be calculated between predicted and historical OD vectors.

3.3.1 Alternative Formulation: Matching Counts Exactly

Since DynaMIT is real-time traffic prediction software, computational efficiency is always an issue, it is important to explore any simplifications to the OD estimation algorithm that could bring the benefit of reduced execution time. Then, one can weigh the benefit of faster execution against the reduced accuracy of the simplified estimator. Thus we motivate this section, which explores a potentially useful simplification.

Another way of setting up the model is to view the indirect measurements as being absolute constraints; in other words, we require that the assigned estimated OD flows exactly match the sensor counts, and minimize the difference between the estimate and the historical flows (or previous estimate). A closed-form estimator is readily available (Kay, 1993) if equality constraints are linear. Here, we require

$$A_h^h x_h = (y_h - p_h) \tag{3.13}$$

and the objective is to minimize

$$J(x) = ||x_h^H - \hat{x}_h + \partial x_h||^2$$

or, with an error covariance matrix,

$$(x_h^H - \hat{x}_h + \partial x_h)' Q_h^{-1} (x_h^H - \hat{x}_h + \partial x_h).$$

where $\partial x_h = \sum_{p=\max(h-d_a,1)}^{h-1} F_h^p (x_p^H - \hat{x}_p)$. The following derivation, however, is based on an error covariance matrix $\frac{1}{\sigma^2}\mathbf{I}$, for simplicity. The GLS version is stated as well. To find the LS estimator we use the method of Lagrange multipliers. We instead minimize

$$L(x, \lambda) = (x_h^H - \hat{x}_h + \partial x_h)'(x_h^H - \hat{x}_h + \partial x_h) + \lambda'[A_h^h \hat{x}_h - (y_h - p_h)]$$

where $\lambda \in \mathbb{R}^{n_l}$ is a vector of Lagrange multipliers, subject to no constraints. The idea here is instead of enforcing the hard equality constraints, we allow them to be violated. However, we associate a Lagrange multiplier, or price, incurred for the violation of each constraint. With the right choice of prices (also called the *dual* vector) the solution to the unconstrained optimization is the same as that of the constrained problem.

The expression above is expanded to

$$L(x, \lambda) = (x_h^H + \partial x_h)'(x_h^H + \partial x_h) - 2\hat{x}_h x_h^H + \hat{x}_h' \hat{x}_h + \lambda' A_h^h \hat{x} - \lambda'(y_h - p_h)$$

Taking the gradient with respect to $\hat{x}_h$ gives

$$\frac{\partial L}{\partial \hat{x}_h} = -2(x_h^H + \partial x_h) + 2\hat{x} + (A_h^h)'\lambda$$

Setting this equal to 0 gives

$$\hat{x}_h = x_h^H + \partial x_h - \frac{1}{2}(A_h^h)'\lambda$$

To find λ we must re-impose the constraint of equation 3.13. Multiplying both sides by A_h^h gives

$$A_h^h \hat{x}_h = (y_h - p_h) = A_h^h(x_h^H + \partial x_h) - \frac{1}{2}A_h^h A_h^{h'}\lambda$$

and so

$$\frac{1}{2}\lambda = (A_h^h A_h^{h'})^{-1}[A_h^h(x_h^H + \partial x_h) - (y_h - p_h)].$$

Therefore our estimator is

$$\hat{x}_h = x_h^H + \partial x_h - A_h^{h'}(A_h^h A_h^{h'})^{-1}[A_h^h(x_h^H + \partial x_h) - (y_h - p_h)] \tag{3.14}$$

The GLS version, from a parallel derivation, is

$$\hat{x}_h = x_h^H + \partial x_h - Q_h^{-1}A_h^{h'}(A_h^h Q_h^{-1} A_h^{h'})^{-1}[A_h^h(x_h^H + \partial x_h) - (y_h - p_h)]$$

This is called the "Exact-Match" estimator.

Note that as discussed before, the matrix $(A_h^h A_h^{h'})^{-1}$ will in general not be invertible, since there could be sensors which do not record and therefore introduce a row of zeros into the assignment matrix. This problem can be solved by removing that sensor, and hence that row of the assignment matrix, for the calculation.

Also, if the number of OD pairs is less than the number of sensors, the matrix $(A_h^h A_h^{h'})^{-1}$ will certainly not be invertible, and this formulation cannot be used. Since this formulation requires that the assigned estimated flows match the sensor counts exactly, if the system is overdetermined this will not be possible.

This estimator is more computationally efficient than the GLS estimator described above. This estimator requires the inversion of a much smaller matrix, one that is $n_l \times n_l$ rather than $(n_{OD} + n_l) \times (n_{OD} + n_l)$. It does require 3 matrix-matrix multiplications as opposed to 2 for the GLS, but these multiplications are on much smaller matrices.

The drawback to this method is that it assumes that the assignment matrices are perfect, and that the estimate of cars on the network which have departed during the current time interval, $y_h - p_h$, is accurate as well. There is no tunability in this method whereby we can give more or less credence to the historical database. Instead, here, the historical database is necessarily given less priority than the sensor constraints, which are satisfied exactly.

3.3.2 Kalman Filter Formulation

Using the Kalman filter methodology of section 2.3.1, we can develop a Kalman filter algorithm for computing the OD vector estimates. As before, we create augmented vectors $\mathcal{Y}_h$ and augmented matrices $\mathbf{A}_h$ which can be used directly in the Kalman filter equations. The initial covariance matrix $\Lambda_{0|-1}$ is set to $\mathbf{C}_0$ and the initial estimate is set to $\hat{x}_0$.

What remains to be determined are the transition matrices. We require that the transition equation be of the same form as the basic state-space transition equation of equation 2.5. From equation 3.4, we can write

$$\hat{x}_{h+1} = F_{h+1}^h \hat{x}_h - F_{h+1}^h x_h^H + x_{h+1}^H + \sum_{p=h+1-d_a}^{h} F_{h+1}^p (\hat{x}_p - x_p^H) + w_h$$

which is of the same form; a transition matrix times the previous estimate plus a constant

$$u_h = -F_{h+1}^h x_h^H + x_{h+1}^H + \sum_{p=h+1-d_a}^{h} F_{h+1}^p (\hat{x}_p - x_p^H)$$

The modified Kalman filter algorithm is then

1. Compute the Kalman gain matrix:

$$K_h = \Lambda_{h|h-1}\mathbf{A}_h'(\mathbf{A}_h\Lambda_{h|h-1}\mathbf{A}_h' + \frac{1}{w_1}\mathbf{R}_h)^{-1}$$

2. Generate the filtered estimate and its associated error covariance:

$$\begin{aligned}\hat{x}_{h|h} &= \hat{x}_{h|h-1} + K_h(\mathcal{Y}_h - u_h - \mathbf{A}_h\hat{x}_{h|h-1}) \\ \Lambda_{h|h} &= \Lambda_{h|h-1} - K_h\mathbf{A}_h\Lambda_{h|h-1}\end{aligned}$$

3. Generate the next estimate and its associated error covariance[6]

$$\begin{aligned}\hat{x}_{h+1|h} &= F_{h+1}^h\hat{x}_{h|h} + u_h \\ \Lambda_{h+1|h} &= F_{h+1}^h\Lambda_{h|h}{F_{h+1}^h}' + \frac{1}{w_2}\mathbf{Q}_h\end{aligned}$$

3.3.3 Choice of Estimator

The main advantage of the Kalman filter formulation is that the entire record of the system's behavior is taken into account when generating each new estimate. This behavior is incorporated into the construction of each additional covariance matrix. The GLS formulation, on the other hand, provides the next estimate looking only at the current sensor data and an autoregressive process on the d_a previous estimates. The estimates provided using the GLS approach presented here, while not the most statistically efficient, may still be acceptable.

Another advantage of the Kalman filter algorithm is that K_h, the Kalman gain matrix, and the error matrices can theoretically be precomputed to save on real-time computations. Unfortunately, in our case, we rely on updated data in order to

[6]Normally, in a Kalman filter algorithm, the "noise" processes v and w are expected to be zero-mean. However, we now introduce a bias corresponding to deviations in previous estimates. This does not affect the covariance update though.

estimate the assignment matrices A_k^h. Thus, using the Kalman filter algorithm in our case can actually be detrimental, since it requires the computation of the covariance matrices as well as the estimates. We have no use for the covariance matrices in themselves, in that they are only used for the internal operation of the filter. In our formulation we can select the covariance matrix to be diagonal, further saving on computations, while in the Kalman filter algorithm, every covariance matrix after the first will necessarily be full.

The Kalman filter algorithm could be computationally more efficient if we agree to sacrifice on the accuracy of the assignment matrices. Instead of using our ad hoc assignment matrix estimator (based on today's OD estimates) we could periodically update the Kalman gain and covariance matrices off-line, or in parallel.

In the sections that follow, several enhancements to the OD estimation model are presented. These enhancements are discussed in terms of a GLS estimator formulation for convenience, but they can just as easily be incorporated into an enhanced Kalman filter using the framework of the previous section. One exception is the constrained estimation. Further investigation is necessary to derive a constrained Kalman filter formulation.

3.4 Enhancements to the Model

We have presented an OD estimation model which results in an estimate which both:

- agrees with the measured sensor counts, and
- makes use of the historical structure of OD flows.

There are other measurement sources and objectives which can be included in the model, and those are explored in this section. The philosophy here is that the basis of

a good OD estimation algorithm is agreement with sensor counts, which is the most accurate set of measurements available. The sensor counts are physically measured quantities that are collected in real time, as opposed to previously estimated OD vectors or OD vectors based on surveys. Any enhancements to our model must use this philosophy as a basis from which to build. A caveat is always that: **while adding more measurements and objectives to the estimator makes the model more robust, there is always a price to pay in terms of increased computation time.**

Our tool for deciding which objective is relatively more important is W, the weighting matrix in the GLS estimator. By adjusting the values on the diagonal of W, we are stating which set of measurements we have more confidence in. For example, in the basic model, if we are not very confident in the historical OD vectors then we would set the entries $W(i,i)$, $i < n_l$, which correspond to the sensor counts, to be higher than the entries for $i > n_l$, which correspond to the deviations objective. If we were more confident in one sensor's accuracy, then its individual entry could be set higher.

This can also be explained from a probabilistic perspective, if we recall the measurement equation with noise, equation 3.1. Each component of the noise vector $v_h(k)$ expresses "noise," or inaccuracies present in measurement k. If we assume each element of the noise vector to be a zero-mean random variable, then each has a variance σ_k^2. If the noise for one element has larger variance than another, it means we should have less confidence in its corresponding measurement. Also, if we further assume that the noise (errors) is uncorrelated from measurement source to measurement source, then the covariance matrix C of the noise vector will be diagonal, since only the variance entries are present.

3.4.1 OD Flow Proportions

It has been observed in Ashok (1996) that the "shares," or the relative proportions of flows from a single origin to various destinations, will remain stable over the course of a day, even as the overall demand fluctuates such as during rush hour. This suggests another objective to add to the estimator in our model. It is debatable which objective is more important: agreement with historical OD flows or share stability. Certainly we would like our OD estimator to be able to handle unforeseen spikes in demand, such as for a special event or incident. Such an event might not be accounted for in the historical database, so just using the historical database as an objective would be misleading. Adding the share fractions allows for unlimited scaling of demand in the real network, since the sensor measurements should indicate the total demand objective required for the estimator. In the following discussion, three variants of the proportions estimator are presented.

A straightforward formulation is as follows. Suppose we focus on the OD pairs with origin i. Suppose there are d such OD pairs, which are numbered sequentially. Then we would desire that the ratio of an OD flow to the total flow from an origin remain constant over time. This ratio would be extracted from the historical database– just as a historical database of absolute flows can be calibrated, so too can a database of flow proportions. The objective is then, for each $j \in (1, \ldots, d)$ that

$$\frac{x_h^H(i \to j)}{\sum_{k\in[1,d]} x_h^H(i \to k)} \approx \frac{\hat{x}_h(i \to j)}{\sum_{k\in[1,d]} \hat{x}_h(i \to k)}$$

remain constant. Rearranging, we have

$$[\psi_h(i \to j) - 1]\hat{x}_h(i \to j) + \psi_h(i \to j) \sum_{k\in[1,d],\, k\neq j} \hat{x}_h(i \to k) \approx 0$$

where ψ_h is a vector of flow proportions, with each element defined as

$$\psi_h(i \to j) = \frac{x_h^H(i \to j)}{\sum_{k\in(1,\ldots,d)} x_h^H(i \to k)}$$

These objectives can be added to the estimator realizing that a column of zeros will be concatenated to the end of the $\mathcal{Y}_h$ measurement vector, and n_{OD} more rows will be added to the observation matrix $\mathbf{A}_h$, of the following form:

$$\begin{bmatrix} & & \vdots & & \\ \cdots & \psi_h(i \to j) - 1 & \psi_h(i \to j) & 0 & 0 & \cdots \\ \cdots & \psi_h(i \to k) & \psi_h(i \to k) - 1 & 0 & 0 & \cdots \\ \cdots & 0 & 0 & \psi_h(m \to n) - 1 & \psi_h(m \to n) & \cdots \\ \cdots & 0 & 0 & \psi_h(m \to o) & \psi_h(m \to o) - 1 & \cdots \\ & & \vdots & & & \end{bmatrix}$$

Here, we have assumed for simplicity (although this is not necessary) that OD pairs from the same origin are grouped together. For a given origin i which has d destinations, we add d rows. Recall that each column of $\mathbf{A}_h$ corresponds to an OD pair, and so the entries for origin i's rows will be zero, except for the columns corresponding to origin i. For the row corresponding to OD pair $(i \to j)$, column $(i \to j)$'s entry is $\psi_h(i \to j) - 1$, while the entries corresponding to origin i's other OD pairs are $\psi_h(i \to j)$.

A second formulation calls for the ratios between pairs of flows from the same origin remain the same. We would like for each $j, k \in (1, \ldots, d)$ that

$$\frac{x_h^H(i \to j)}{x_h^H(i \to k)} \approx \frac{\hat{x}_h(i \to j)}{\hat{x}_h(i \to k)}$$

Rearranging, we have

$$\hat{x}_h(i \to k)\frac{x_h^H(i \to j)}{x_h^H(i \to k)} - \hat{x}_h(i \to j) \quad \approx \quad 0$$

These objectives can be added to the estimator realizing that a column of zeros will be concatenated to the end of the $\mathcal{Y}_h$ measurement vector, and more rows will be added to the observation matrix $\mathbf{A}_h$. Each row will be all zeros, except for two

entries, $(x_h^H(i \rightarrow j)) \div (x_h^H(i \rightarrow k))$ at the column for OD pair $(i \rightarrow k)$ and -1 at the column for OD pair $(i \rightarrow j)$.

The drawback is that the dimension of the estimator is now quite large. Focusing on a single origin, suppose that k unique OD pairs are active. Then we must add $\binom{k}{2}$ additional lines to the augmented assignment matrix. This number increases even faster than $n(n-1)$, which is the maximum number of OD pairs. Still, we do not expect that every possible OD pair for a given active origin will be active, and so in many cases this objective can be added to the model with an acceptable increase in computing time.

A third idea is for the ratio of the total flow from one origin to another origin remain constant. By itself, this objective allows variation in the ratios of individual flows from the same origin, but this objective can be combined with one of the first two mentioned above, if both types of proportions can be calibrated. The objective is

$$\frac{\sum_{k\in[1,d_i]} x_h^H(i \rightarrow k)}{\sum_{k\in[1,d_j]} x_h^H(j \rightarrow k)} \approx \frac{\sum_{k\in[1,d_i]} \hat{x}_h(i \rightarrow k)}{\sum_{k\in[1,d_j]} \hat{x}_h(j \rightarrow k)}$$

where we assume that there are d_i possible destinations for origin i and d_j possible destinations for origin j. This is again linear in the variables to be estimated, and the estimator can be augmented accordingly. Here, we require that $\binom{n_O}{2}$ rows be added to the observation matrix $\mathbf{A}_h$ and $\binom{n_O}{2}$ zeros be concatenated to $\mathcal{Y}_h$, where n_O is the number of origins.

Among these three formulations of the proportions objective, several comparisons can be made. Since the calibration of the proportions is done off-line, each formulation is equivalent in terms of the preparation of the observation matrices. The third is the least computationally intensive, but provides the least information about the flows. The second is the most computationally intensive, adding the most to the dimension of the estimator, and provides the most information about the relationships of the flows

to each other. The first formulation strikes a balance between statistical efficiency and computational efficiency. Ashok (1996) has demonstrated the validity of the objectives of the first type, and further research is necessary to verify the others' validity and usefulness.

3.4.2 Probe Vehicles

A very promising development in the ITS field is the increasing possibility of using *probe vehicles* to provide direct measurements of OD flows. A probe vehicle is a vehicle equipped with a device that enables a traffic management center to track its progress through the network. If probe vehicles are a statistically significant portion of the driver population, we can then estimate that

$$\hat{x}_h = \frac{1}{f_h} x_h^{\text{probe}}$$

where f_h is the percentage of the drivers which are probe vehicles. To express this in the LS estimation context, we write

$$x_h^{\text{probe}} = F_h x_h + \epsilon_h$$

where F_h is a diagonal matrix whose diagonal entries are the percentages of drivers for each OD pair which are probe vehicles. ϵ_h indicates error in this formulation. Of course, this assumes that a calibrated estimate of F_h is available. It is straightforward to incorporate these additional direct measurements by following a procedure like the one above for augmenting the least squares formulation

$$\mathcal{Y} \approx \mathbf{A}\mathcal{X}.$$

The data from probe vehicles can also be used to refine the estimation of the assignment matrices. Estimation of the assignment matrix currently relies on the

supply simulator to give an estimate of the travel times in the network, which could be improved by the sample of travel times provided by the probe vehicles.

One exciting application of probe vehicle technology is being implemented in Westchester County, New York by the New York State Department of Transportation (NYSDOT). In the New York City area, there is a high penetration rate of the EZ-Pass electronic toll collection (ETC) system. According to the Port Authority of New York and New Jersey, over 60% of drivers used ETC at its bridges and tunnels during off-peak hours, which increased to over 75% during peak hours. With this in mind, NYSDOT is in the process of installing 24 EZ-Pass detectors at locations throughout the Westchester County network. A similar array of detectors has already been installed along the Garden State Parkway and New York State Thruway near the New York / New Jersey border. Guidance generation systems such as DynaMIT are particularly suited to the Westchester network, which is shown schematically in figure 3-4. With 5 or more major north-south routes available to passenger cars, and 2 major east-west connectors, there are many feasible paths for travelers through the area.

3.4.3 Constrained Estimation

Nonnegativity Constraints

The previous presentation has ignored the fact that the OD flows may not take on any value. Most obviously, no OD flow can be negative. It has been found that given the parameters of the OD estimation problem, there are cases where the algorithm will estimate an OD vector with negative entries. This is more likely to happen if some OD flows are very small compared to others, since we expect reasonable variation around the true flow.

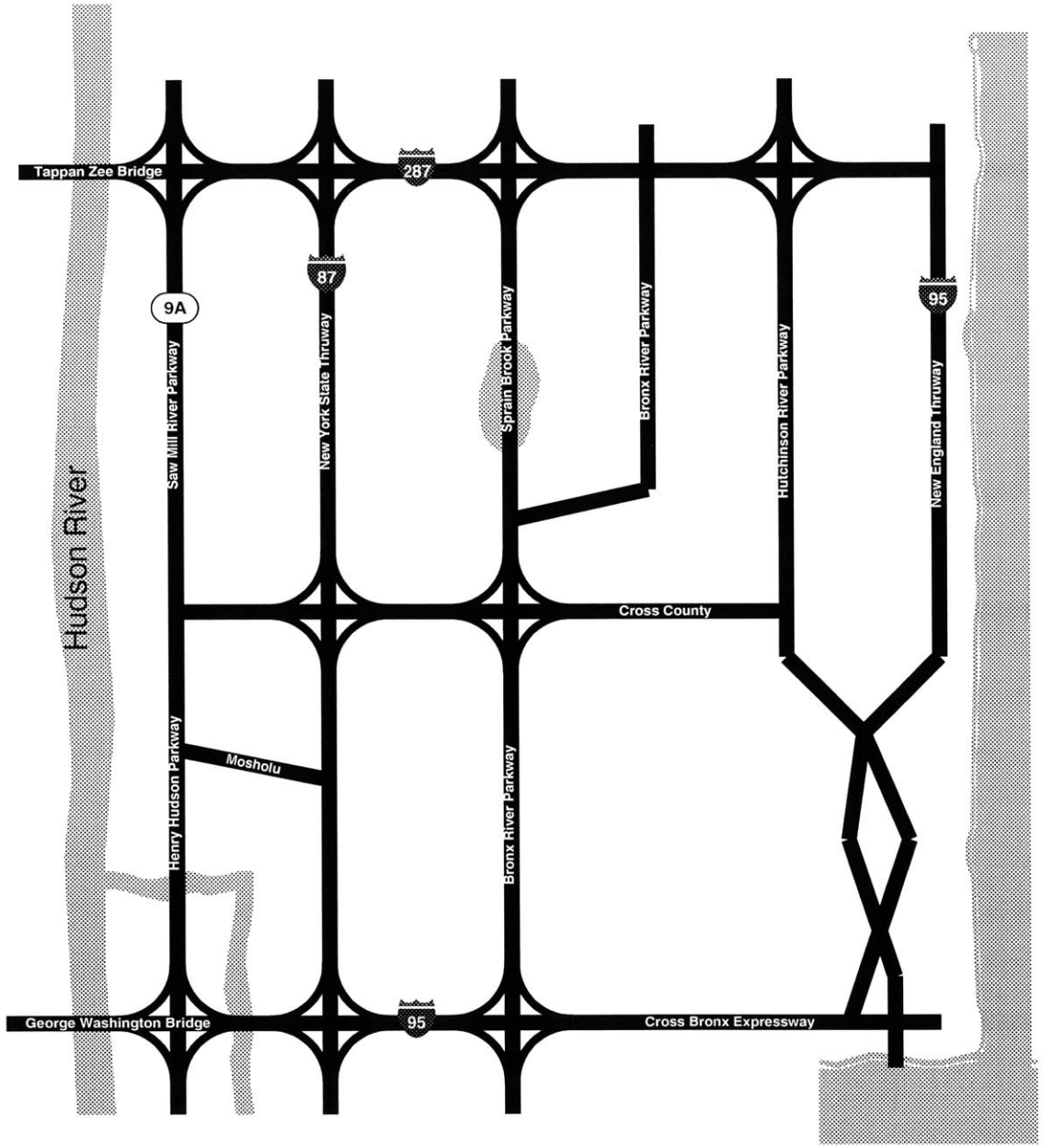

Figure 3-4: Schematic of Westchester County highways.

Lawson and Hanson (1974) present an algorithm for the NNLS (nonnegative least squares) estimation problem, defined as in equation 2.2. Let x have n elements, and let A be a $m \times n$ matrix, so that y has m elements.

An optimal solution to the NNLS problem must satisfy what are known as the Kuhn-Tucker optimality conditions. A vector $\hat{x}$ is a solution if and only if there exists a vector $\hat{f}$ and a partitioning of the integers 1 through m into subsets $\mathcal{E}$ and $\mathcal{S}$ such that

$$\hat{f} = A'(A\hat{x} - y)$$

$$\hat{x}(i) = 0 \text{ for } i \in \mathcal{E}, \qquad \hat{x}(i) > 0 \text{ for } i \in \mathcal{S}$$

$$\hat{f}(i) \geq 0 \text{ for } i \in \mathcal{E}, \qquad \hat{f}(i) = 0 \text{ for } i \in \mathcal{S}$$

The vector $\hat{f}$ is known as the dual vector. These conditions give the basic idea for most available solution algorithms: An optimal partitioning of the components of the estimate is determined. In one partition, the variables can be optimized in the least squares sense, and in the other partition, the variables must be set to zero.

For the NNLS algorithm, we define index sets $\mathcal{Z}$ and $\mathcal{P}$, which list the variables that are set to 0 and those that are free, respectively. At each major iteration of the algorithm, if a variable in the set $\mathcal{P}$ becomes negative, it will be forced positive or transferred to the set $\mathcal{Z}$. Please see Appendix B for the algorithm as presented by Lawson and Hanson.

One ad hoc method for preventing negative OD flows is to simply set those flows which have been estimated to be negative to zero, or to some small positive value. This will result, however, in the estimate being suboptimal in the data available. If it is determined that the performance of the estimator using the ad hoc method is unacceptable, then one of the constrained LS estimation methods must be used.

The ad hoc method described essentially chooses the partitioning by assigning every variable that has come out negative to the set of variables that is set to zero.

Capacity Constraints

Another possibility is that an estimate is produced which has the total flows from a certain origin which exceed the input capacity of that node. The input capacity is used by the supply simulator to restrict the number of vehicles allowed to enter the network per time period. This would suggest constraints of the following form:

$$\sum_{k|(i\rightarrow k)\in\mathcal{OD}} \hat{x}_h(i \rightarrow k) \leq c_i \tag{3.15}$$

Violation of these constraints could potentially cause serious problems in DynaMIT's OD estimation system due to the iterative process used to reach consistency between the estimated OD vector and the assignment matrix. As explained earlier in section 3.1.2, entries $A_h^h(i,j)$ in the assignment matrices are obtained by dividing the number of vehicles counted on sensor i from OD pair j by the total OD flow $x_h(j)$. It is always possible for a sensor to detect a larger number of vehicles in real life than is anticipated by the historical OD vectors and the supply simulator. For instance, if we have been consistently underestimating the demand for an OD pair, there may be drivers from that pair which have departed in previous time intervals which are not subtracted as part of the p_h vector. If the supply simulator is miscalibrated, it could be simulating unrealistic congestion at an earlier point along the drivers' paths causing fewer drivers to be simulated at the sensor point. If the route choice models are faulty, then DynaMIT might simulate fewer drivers from other OD pairs crossing the sensor, while in fact more take that path in real life. Additionally, it has been observed that the solution to the LS problem may exceed what would be expected simply because this, in combination with all of the other variables, minimizes the

given objective function.

Continuing with the example of the Florian network, suppose loader node 1 has an input capacity of 2000 vehicles. Suppose for simplicity that only a single OD pair is active. Suppose that while the surveillance system shows 2500 vehicles crossing the sensor on link 9, the historical database shows that only a demand of 2000 is expected on OD pair (1,4). The assignment matrix estimated the first time the supply simulator is run is $A_1^1 = [2000/2000] = [1]$. The estimate computed will be $\hat{x}_1^1 = 1 \times 2500 = 2500$ vehicles per period. Then, as the first estimate is fed back into the supply simulator to iterate and reestimate the assignment matrix, the demand of 2500 exceeds the input capacity, and only 2000 vehicles are simulated to cross the sensor. The next assignment matrix, $A_1^2 = [2000/2500] = [0.8]$. So, instead of further iterations refining both the assignment matrix and OD estimate, the process diverges, producing the estimates shown in table 3.1.

Iteration	Assignment Matrix	OD Estimate
1	[1]	2500
2	[0.8]	3125
3	[0.64]	3906
⋮	⋮	⋮
10	[0.086]	18626

Table 3.1: Unstable iterations of demand estimation.

This instability is due to the presence of capacity constraints in the supply simulator, but a lack of the same constraints in the demand simulator. There are several possible methods for controlling this instability. The simplest is to not allow any estimated flows to exceed loader input capacities by adjusting the OD flows from that loader to be lower. This would be done while keeping the relative proportions

of those OD flows the same. So if for some loader i we have

$$\sum_{k|(i\rightarrow k)\in\mathcal{OD}} \hat{x}_h(i \rightarrow k) > c_i$$

we would adjust the flows to

$$\hat{x}_h^{\text{adj}}(i \rightarrow j) = \frac{\hat{x}_h(i \rightarrow j)}{\sum_{k|(i\rightarrow k)\in\mathcal{OD}} \hat{x}_h(i \rightarrow k)} \times c_i$$

However, as mentioned in the discussion of the nonnegativity constraints, simply truncating the estimated flows could result in a suboptimal solution. This would suggest the use of an LSI (least squares with inequality constraints) algorithm using the constraints of equation 3.15 and possibly the nonnegativity constraints as well. In matrix form, the capacity constraints would be written

$$Cx \geq b$$

where $-b$ is the vector of loader capacities and $C \in \mathbb{R}^{n_O \times n_{OD}}$ is the capacity mapping matrix, and O is the number of origin loaders. C will be of the form

$$C = -\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & \dots & 0 \\ & & & & \vdots & & & \end{bmatrix}$$

Constraints of this form can be used with the algorithms such as the one described in Lawson and Hanson (1974) which involves a transformation of the LSI problem into an LDP (least distance programming) problem which can then be solved using a variant of the NNLS algorithm. The drawback of this method is that it is computationally intensive, since all solutions to the LSI problem involve iterative convergence to a solution.

The following discussion describes how to transform the OD estimation problem with capacity (and nonnegativity) constraints into a form that can be solved using

an LDP algorithm. One must first obtain the orthogonal transformation (by using a singular-value decomposition (SVD) algorithm) of the matrix

$$A = [Q_1 \quad Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} K'$$

where Q is $m \times m$ and orthogonal, K is $n \times n$ and orthogonal, and R is $n \times n$ and full rank. Q can be partitioned into $Q_1 \in \mathbb{R}^{m \times n}$ and $Q_2 \in \mathbb{R}^{m \times m-n}$. We then introduce the change of variables

$$x = Kw.$$

We can then write, since multiplying the objective function by a constant matrix Q does not change the optimal solution,

$$\begin{aligned} Q'(y - Ax) &= Q'y - Q'[Q_1 \quad Q_2] \begin{bmatrix} R \\ 0 \end{bmatrix} K'Kw \\ &= Q'y - \begin{bmatrix} Rw \\ 0 \end{bmatrix} \end{aligned}$$

since K and Q are orthogonal, and therefore $K'K = Q'Q = \mathbf{I}$. The objective function to be minimized becomes

$$\begin{aligned} \min ||y - Ax|| &= \left|\left| \begin{bmatrix} Q_1'y \\ Q_2'y \end{bmatrix} - \begin{bmatrix} Rw \\ 0 \end{bmatrix} \right|\right| \\ &= ||Q_1'y - Rw|| + ||Q_2'y|| \end{aligned}$$

If we let $z = Q_1'y - Rw$, and realizing that minimizing $||z||$ plus a constant is equivalent to minimizing $||z||$ alone, we have the LDP problem

$$\begin{aligned} \min \quad & ||z|| \\ \text{subject to} \quad & Cx \geq b \end{aligned}$$

We then have $Cx = CKw$, and $w = R^{-1}(z - Q_1'y)$, and then multiplying this out gives the LDP problem

$$\begin{aligned} \min \quad & ||z|| \\ \text{subject to} \quad & CKR^{-1}z \geq b - CKR^{-1}Q_1'y \end{aligned}$$

Lawson and Hanson give an algorithm for solving the LDP problem, which can be found in Appendix B. The optimal OD vector x can be found using the transformation $x = KR^{-1}(z - Q_1'y)$.

3.4.4 Holding Small Flows Fixed: Large-Flow Estimator

DynaMIT estimates the OD flow for each OD pair present in the historical database. It is possible that many OD pairs have very small demand relative to other OD pairs. The presence of these OD pairs in the problem has several adverse effects on the estimation problem, which will be described below. If these small-flow OD pairs could be removed from the estimation, an immediate benefit would be a reduction in computation time. Of course, these OD flows are still present in the network, contributing to sensor counts. In order to accommodate their presence in the sensor count totals, the measurement equation 3.8 should be modified to

$$y_h - p_h - s_h = \tilde{A}_h^h \tilde{x}_h + \tilde{v}_h$$

where s_h is a vector of the sensor counts due to the vehicles from small-flow OD pairs. The assignment matrix and OD vector are written with tildes to indicate that these do not contain entries corresponding to the small-flow pairs. $\tilde{A}_h^h$ can be formed by deleting the columns corresponding to the small-flow OD pairs. s_h can be computed in the same way that p_h is, since the DynaMIT supply simulator keeps track of all simulated vehicles according to their OD pairs.

The presence of the small-flow OD pairs causes several problems with the OD estimation algorithm, which are also explored in the case studies. Firstly, it can be argued that measurements for a single time interval (of, for example, 15 minutes) cannot give useful information about small flows. In other words, such small flows might not be *observable*. The problem is quite sensitive to inaccuracies in the supply simulator and route choice model. If in the assignment matrix estimation phase of DynaMIT it happens that while we expect 4 vehicles to be detected by a certain sensor, only 1 actually is, the entry in the assignment matrix would be significantly different. The estimate $\hat{x}_h(j) = x_h^H(j)$ would be more reasonable in this case.

It has been observed experimentally that negative OD flows are more likely to be estimated when there are some small-flow OD pairs in the historical database. Since we have to either set the negative flows to zero, causing the solution to be perhaps suboptimal, or reestimate using a computationally intensive NNLS algorithm, this is a serious nuisance. Taking the small-flow OD pairs out of the formulation would solve these problems, and only involve some easy computations to determine s_h.

Chapter 4

Case Studies

To demonstrate the correctness and effectiveness of the OD estimation model presented above, a number of tests have been performed using three networks: The Florian network of figure 4-1, the Central Artery / Tunnel (CA/T) network in Boston, Massachusetts, and the Irvine, California, ITS testbed. The networks present issues of increasing complexity due to the number of OD pairs, available paths, and more complicated traffic dynamics.

The objective of the experiments is to:

- Verify the accuracy of the basic GLS formulation with the simplification that requires only the current time interval's OD vector be estimated.
- Evaluate the performance of the Exact-Match estimator of section 3.3.1.
- Evaluate the loss of accuracy incurred when holding small flows fixed, and weigh this against the benefit of reduced computation time.
- Determine the benefits of using OD flow proportions in the GLS estimator, and weigh these benefits against the increase in computations required.

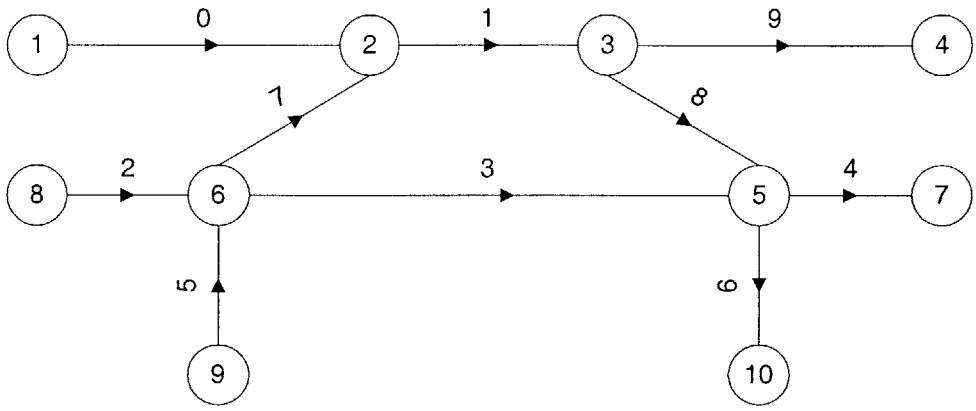

Figure 4-1: The "Florian" network.

- Evaluate the need for introducing nonnegativity and / or capacity constraints, and describe the situations in which violations of these constraints are likely to occur.

Sensor counts were generated using MITSIMlab, a microscopic traffic simulator, described in Yang and Koutsopoulos (1996), using the historical or synthetic OD flows available. MITSIMlab can simulate traffic at a high level of detail. The network file, which is the same format for DynaMIT and MITSIMlab, can represent individual lanes, ramps and toll plazas. Different vehicle types can be simulated, including cars, trucks and buses. Varying driver behavioral characteristics, such as speed preference and lane-changing behavior, can be simulated.

As a general note, the accuracy of the OD estimation depends heavily on the accuracy of the assignment matrix. The accuracy of the assignment matrix is influenced by the route choice models used to assign vehicles to alternative paths and by the supply simulator used to capture traffic dynamics.

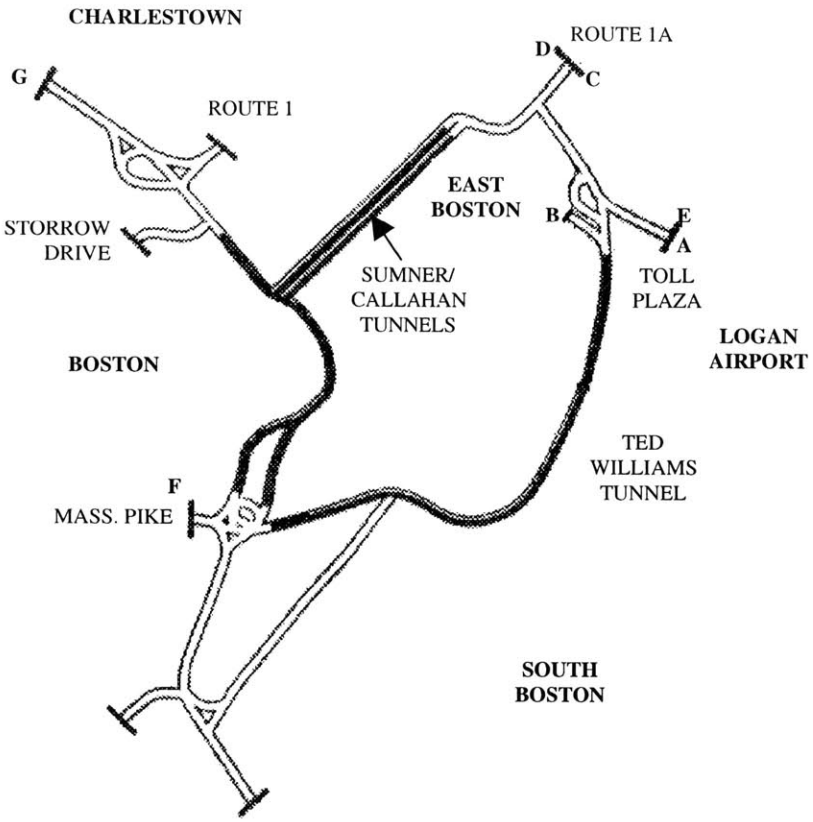

Figure 4-2: Boston's CA/T network.

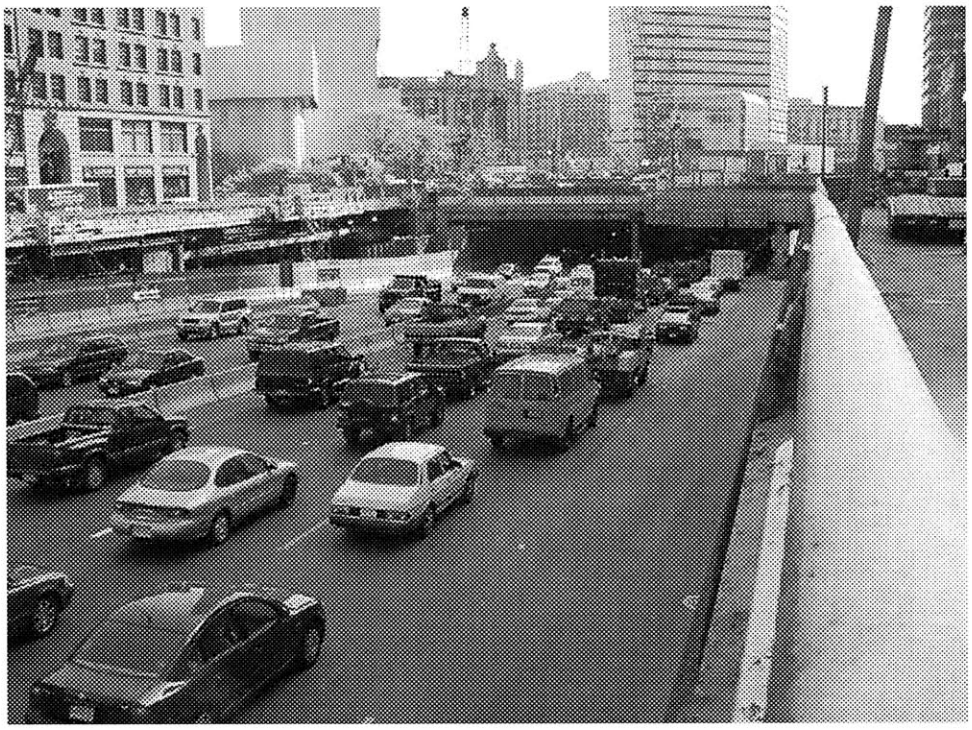

Figure 4-3: A view of the Central Artery.

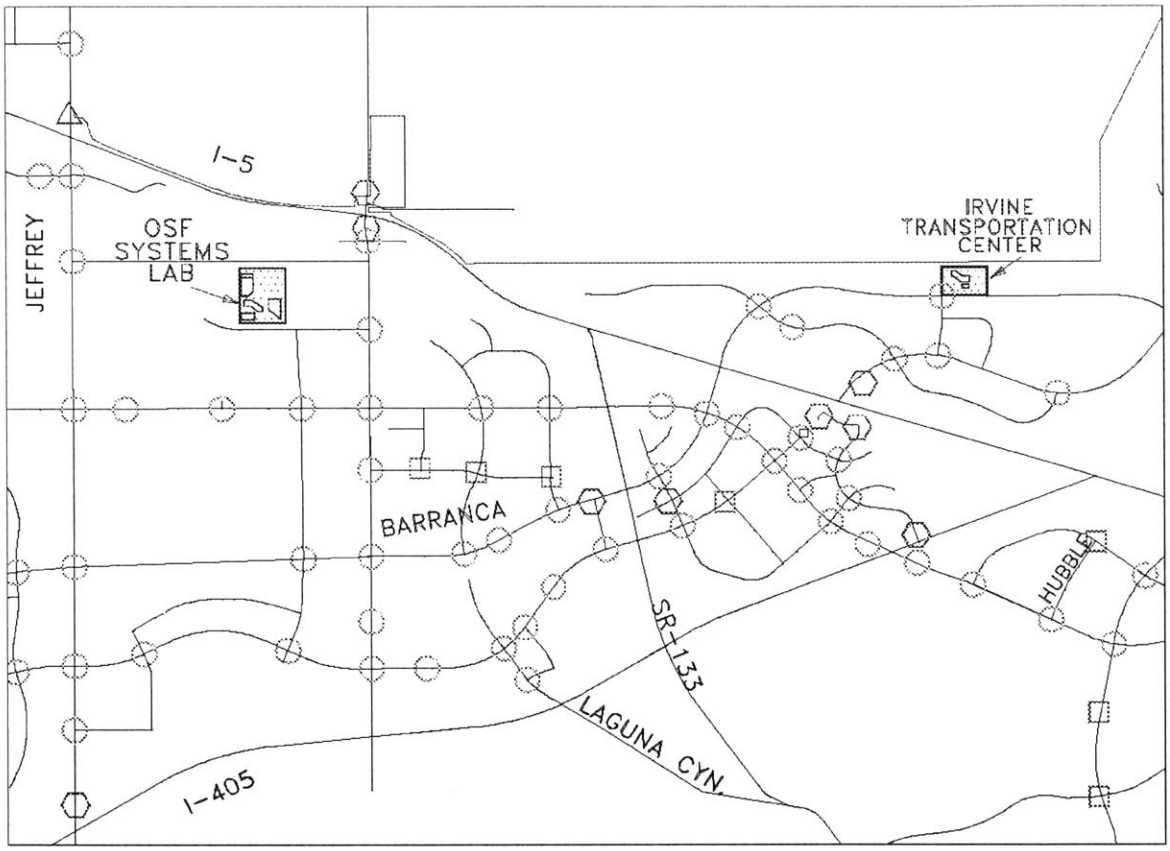

Figure 4-4: The Irvine, California, network. From the City of Irvine Annual Traffic Management Report, 1999–2000.

Figure 4-5: The Irvine network as coded in MITSIMlab.

DynaMIT's supply simulator requires calibration using data from MITSIMlab. In addition, for the output of DynaMIT to be comparable to MITSIMlab's, the route choice models in both should be comparable. As we will see below, the assignment matrices generated by MITSIM and DynaMIT have differences. Therefore, while we do compare the OD matrix used in MITSIM with the output of DynaMIT's OD estimation using DynaMIT's assignment matrix, more meaningful results can be observed when we take MITSIM's assignment matrix as our estimate. This approach is reasonable, since our objective in these case studies is to evaluate how well the different approaches perform.

In order to compare the results of the various approximations and algorithms discussed in chapter 3, we use the following statistics:

- Root mean square (RMS) error:

$$\sqrt{\frac{\sum_j [x_h(j) - \hat{x}_h(j)]^2}{n_{OD}}}$$

- Root mean square, normalized (RMSN) error:

$$\frac{\sqrt{n_{OD} \sum_j [x_h(j) - \hat{x}_h(j)]^2}}{\sum_j x_h(j)}$$

4.1 Networks Considered

Synthetic data was generated for use in the Florian and CA/T networks, while some survey OD data was available for use in the Irvine network.

Based on the OD profile generated, four scenarios are considered:

1. The "true" demand levels are independently perturbed by a random amount uniformly distributed between 95% and 105% of the original demand.

2. The demand levels are perturbed to within 90% and 110% of the original demand.

3. The demand levels are increased by 10%.

4. The demand levels are decreased by 10%.

The transition matrices are assumed here to be a scalar times the identity matrix. For all cases, an autoregressive process of degree 2 is used, so that deviations from

only the prior 2 time intervals are considered in the estimator. The transition matrices are set to $(1/3)\mathbf{I}$, an ad hoc choice. So we assume the transition equation is

$$x_h = x_h^H + \frac{1}{3}\mathbf{I}(x_{h-2} - x_{h-2}^H) + \frac{1}{3}\mathbf{I}(x_{h-1} - x_{h-1}^H)$$

The error covariance matrices are assumed to be diagonal weighting matrices, nominally identity matrices.

The Florian network has already been described. (See figure 4-1.) Nine OD pairs are considered here, with nodes 1, 8 and 9 as the origins and nodes 4, 7 and 10 as the destinations. The surveillance system consists of one sensor per link. The historical tables used as inputs to the OD estimation are generated synthetically for five 15-minute intervals from 7:00 A.M. until 8:15. The peak demand occurs at the third interval (7:30–7:45). The second and fourth intervals have 90% of the peak demand, and the first and fifth have 80% of the peak. The OD matrix for the 9 OD pairs and the 5 time intervals is illustrated in table 4.1. Note that while OD pairs 1–3 have only one path available, the other OD pairs have two paths apiece.

	OD Pair	7:00	7:15	7:30	7:45	8:00
1	$(1 \rightarrow 4)$	320	360	400	360	320
2	$(1 \rightarrow 7)$	320	360	400	360	320
3	$(1 \rightarrow 10)$	320	360	400	360	320
4	$(8 \rightarrow 4)$	240	270	300	270	240
5	$(8 \rightarrow 7)$	240	270	300	270	240
6	$(8 \rightarrow 10)$	240	270	300	270	240
7	$(9 \rightarrow 4)$	280	315	350	315	280
8	$(9 \rightarrow 7)$	280	315	350	315	280
9	$(9 \rightarrow 10)$	280	315	350	315	280

Table 4.1: Generated OD matrix for the Florian network.

The CA/T network consists of several major highways which go through Boston,

including I-93 (the Central Artery), the Sumner, Callahan and Ted Williams tunnels, and the Massachusetts Turnpike (MassPike). This network connects downtown Boston and Cambridge with Logan Airport and MA-1A (East Boston). Parts of this network are currently under construction as Boston's "Big Dig" project, which is expected to be completed by 2004. This network is shown in figure 4-2. This network has 182 nodes and 211 links, and can be considered a network of medium complexity. The surveillance system consists of 35 sensors. We consider five origins and two destinations, for a total of 10 OD pairs.[1] The origins, labeled A through E in figure 4-2, are located in East Boston, while the destinations, labeled F and G. Destination F represents travelers going to the south and west, while destination G represents travelers going to the north and west.

The 10 OD pairs are summarized as follows:

1. $(A \rightarrow F)$ Logan Airport to the MassPike. Two paths are possible, one through the Sumner Tunnel and the other through the Ted Williams Tunnel, which is shorter.

2. $(A \rightarrow G)$ Logan to I-93 North. Two paths are possible: Sumner, which is shorter, or Ted Williams.

3. $(B \rightarrow F)$ Logan to the MassPike. Only one path is possible, since drivers enter directly onto the Ted Williams.

4. $(B \rightarrow G)$ Logan to I-93 North. Only one path is possible, as above.

5. $(C \rightarrow F)$ Route 1A (East Boston) to the MassPike. Two paths are possible, and the Sumner Tunnel is slightly shorter.

[1]The demand profile considered here is the same as for previous tests of DynaMIT using the CA/T network.

6. $(C \rightarrow G)$ Route 1A to I-93 North. Two paths are possible, and the Sumner Tunnel is about 44% shorter.

7. $(D \rightarrow F)$ Similar to OD pair 5, but only the Sumner Tunnel path is available.

8. $(D \rightarrow G)$ Similar to OD pair 6, but only the Sumner Tunnel path is available.

9. $(E \rightarrow F)$ Like OD pair 1, but only the Sumner Tunnel path is available.

10. $(E \rightarrow G)$ Like OD pair 2, but only the Sumner Tunnel path is available.

The historical tables used as inputs to the OD estimation are generated synthetically for five 15-minute intervals from 7:00 A.M. until 8:15. The peak demand occurs at the third interval (7:30–7:45). The second and fourth intervals have 90% of the peak demand, and the first and fifth have 80% of the peak. The OD matrix for the 10 OD pairs and the 5 time intervals can be seen in table 4.2.

OD Pair		7:00	7:15	7:30	7:45	8:00
1	$(A \rightarrow F)$	240	270	300	270	240
2	$(A \rightarrow G)$	240	270	300	270	240
3	$(B \rightarrow F)$	120	135	150	135	120
4	$(B \rightarrow G)$	120	135	150	135	120
5	$(C \rightarrow F)$	180	202.5	225	202.5	180
6	$(C \rightarrow G)$	180	202.5	225	202.5	180
7	$(D \rightarrow F)$	60	67.5	75	67.5	60
8	$(D \rightarrow G)$	60	67.5	75	67.5	60
9	$(E \rightarrow F)$	120	135	150	135	120
10	$(E \rightarrow G)$	120	135	150	135	120

Table 4.2: Generated OD matrix for the CA/T network.

The Irvine network is part of the new ITS testbed in Irvine, California. The network contains the major highways I-5, I-405 and CA-133, as well as all of the

arterial roads, in a triangular area defined by I-5, I-405 and Jeffrey Road. (Please see figure 4-4 for a map of the network.) The Irvine network is quite complex, with 296 nodes and 618 links. The surveillance system consists of 214 sensors which provide information about 626 possible OD flows. The OD flows available were generated from survey data by the California Department of Transportation. The flows available for use span the period from 6:45 A.M. to 7:00, and are static over that period. The complexity of the demand profile of the Irvine network can be seen in figure 4-6, where each active OD pair is indicated by a line drawn from origin to destination.

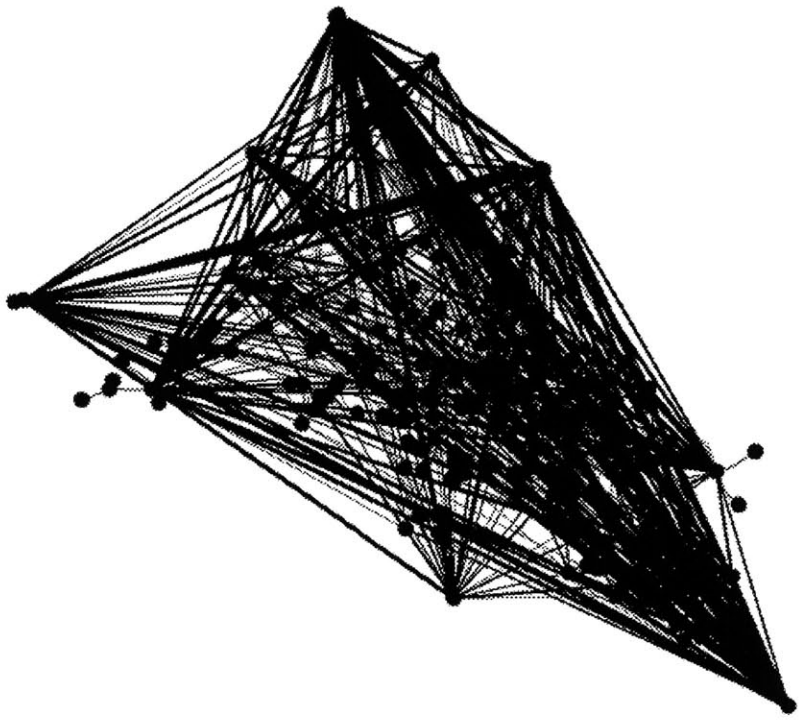

Figure 4-6: Irvine OD schematic.

4.2 Alternative Formulations and Algorithms

4.2.1 Basic LS Estimator

This section describes the performance of the basic OD estimation algorithm, with the simplification described in section 3.3. This estimator uses the sensor counts and deviations from historical flows as its objectives.

The results for the Florian network are shown in table 4.3. For the CA/T network, the results are shown in table 4.4. For the time intervals in question, the errors are given for the estimated OD flows, as well as for the historical flows.

For the second scenario, the errors in the historical flows will increase, as does the estimation error. For the third scenario, in which the "real-life" OD flows have been perturbed to be 10% higher than the historical, as expected, the errors in the historical database will increase. The estimation error does not increase at all. (See figure 4-7 which shows the historical, true and estimated OD flows for the first OD pair, $(1 \rightarrow 4)$ of the Florian network.) The results for the fourth scenario are similar to the third.

For comparison, if the supply simulator assignment matrix estimate is used in time interval 1 of scenario 1 on the Florian network, the RMS error is 4.5470, indicating that more calibration of the route choice models and supply simulator needs to take place.

For the Irvine network, the basic estimator under scenario 3 achieves an RMS error of 92.6938, and an RMSN error of 6.5218. Poor results using the Irvine network can be attributed to the lack of a thorough calibration of the network. It was observed using the MITSIMlab simulator that large queues develop at the network's loaders, with the result that many fewer vehicles than indicated by the historical demand enter the network. This causes errors in the estimation of the assignment matrix and

Scenario	Time Int.	RMS Error Estimate	RMS Error Historical	RMSN Error Estimate	RMSN Error Historical
1	1	1.2906	2.2531	0.0185	0.0324
	2	2.6496	2.6523	0.0345	0.0345
	3	1.9178	1.8911	0.0219	0.0216
	4	2.0841	2.8149	0.0352	0.0353
	5	0.9569	2.1490	0.0138	0.0309
2	1	3.5420	4.7792	0.0510	0.0688
	2	3.8889	4.6800	0.0490	0.0590
	3	2.1085	3.3166	0.0245	0.0385
	4	2.6963	4.5361	0.0351	0.0591
	5	2.4310	4.1433	0.0352	0.0600
3	1	0.9219	7.0475	0.0120	0.0915
	2	0.8055	7.9284	0.0084	0.0915
	3	1.0414	8.8093	0.0108	0.0915
	4	0.6936	7.9284	0.0080	0.0915
	5	0.9172	7.0475	0.0119	0.0915

Table 4.3: Estimation performance in Florian network; basic estimator.

in the simulated vehicle counts.

We also perturb the assignment matrix in the case of scenario 3 of the Florian network, to give an idea of how sensitive the estimation is to assignment matrix errors. Each nonzero entry of the perturbed assignment matrix is randomly chosen from a uniform distribution of the said percentage around the true assignment matrix. The results for these experiments are contained in table 4.5. Only after the assignment matrix has been perturbed by 30% does the estimation error exceed that of the error in the historical OD vector. For scenario 1 the results are different, since there was less leeway between the estimation and historical error to begin with, and the estimate is worse than the historical by the time the assignment matrix is perturbed 15%.

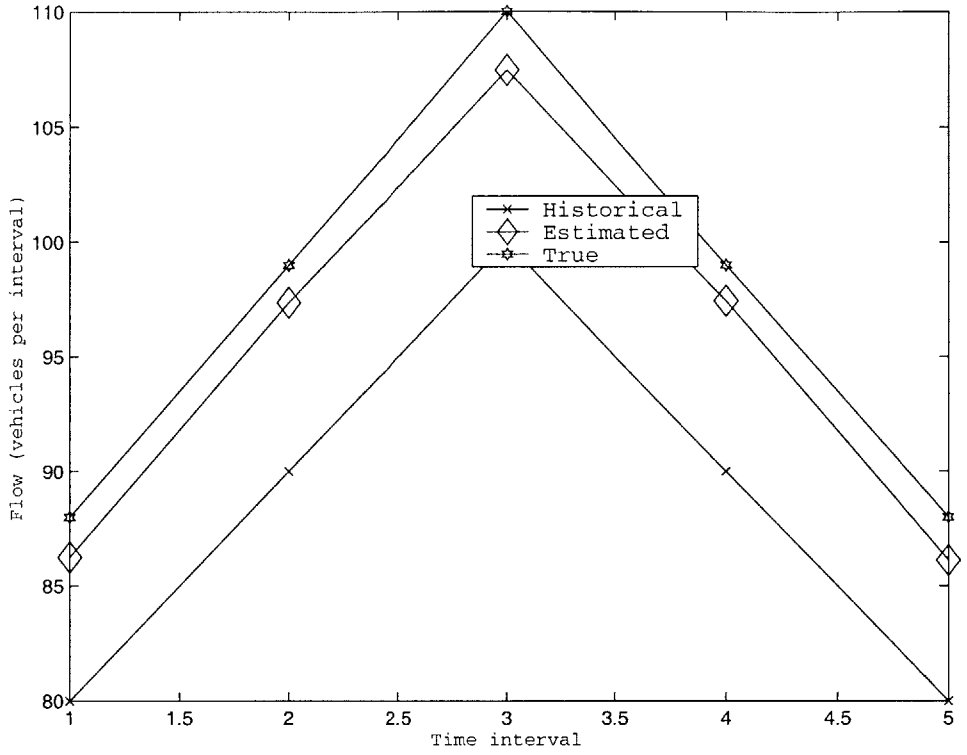

Figure 4-7: Flows for Florian network, OD pair $(1 \rightarrow 4)$.

4.2.2 Exact-Match Estimator

In this section we present results for the Exact-Match estimator of section 3.3.1. This formulation decreases computational complexity by requiring that the sensor counts be matched exactly, and then optimizing to reduce the deviations from the historical database.

Note that a single estimation takes approximately 100×10^3 *flops* (floating point operations) for the CA/T network. The exact-match formulation saves a lot on computation time, taking only 5000 *flops* for an estimation in the CA/T network,

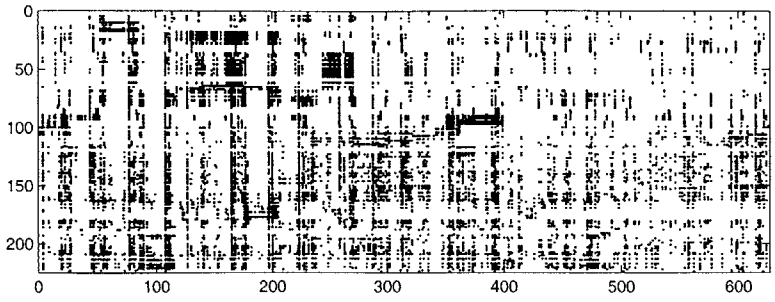

Figure 4-8: Assignment matrix structure for Irvine network. Dark points indicate a non-zero entry.

a savings of a factor of 20 in this case. The basic estimator on the Irvine network requires 500 *Mflops*, where 1 *Mflop* is 1 million *flops*. If one concentrates on just the size of the matrix inversion required, the exact-match formulation has a clear advantage. Recall that the exact-match formulation is only usable if there are more OD pairs than sensors. Matrix inversion is considered to be an $O(n^3)$ operation, where n is the dimension of the matrix. The speed advantage for the exact-match formulation is then $(\frac{n_{OD}+n_l}{n_l})^3$ to 1. So for a case such as the Irvine network, in which there are 626 OD pairs and 214 sensors, the speedup is on the order of 60 times.

A result from the use of the exact-match formulation on the CA/T network is shown in table 4.6. Although for the experiment shown the estimator performance is quite good, this can be considered to be a fortuitous accident caused by judicious selection of the sensors to be considered. In the case of the CA/T network, where there are 35 sensors and only 10 OD pairs, the exact-match formulation is not applicable unless sensor counts are ignored, which would never be done in a real situation. If a different subset of the sensors is considered, the performance is worse, with the estimate's RMS error approaching twice that of the historical error in some cases.

Scenario	Time Int.	RMS Error Estimate	RMS Error Historical	RMSN Error Estimate	RMSN Error Historical
1	2	2.8360	4.8425	0.0198	0.0339
2	1	4.0189	9.3511	0.0475	0.0629
	2	6.2502	10.2777	0.0390	0.0642
	3	7.2354	15.6109	0.0402	0.0867
	4	6.9550	10.9533	0.0435	0.0685
	5	5.5205	11.8089	0.0380	0.0812

Table 4.4: Estimation performance in CA/T network; basic estimator.

The real benefit of using the exact-match formulation is seen using the Irvine network, in which there are many more OD pairs than sensors. To deploy this algorithm in the Irvine network, duplicate sensors must be removed from the measurement equation. A sensor is considered a duplicate sensor if its output is the same as another sensor; or, more generally, if its output is equal to some linear combination of the other sensors' output.

4.2.3 OD Flow Proportions

Testing of an estimation algorithm using OD flow proportions shows promising results on the Florian network. The first variation of the "proportion algorithm" is implemented using only the sensor counts and proportions as targets in the estimator. In the third scenario of demands increased by 10%, for the first time interval we have an RMS error of only 0.1608. For the second time interval the RMS error for the estimate is 0.2556. These results are better than the standard estimator using the deviations as a target, but mostly because the proportions are in fact fixed through each time interval. When the same algorithm is run on the first scenario, in which

Scenario	Amount	RMS Error	RMSN Error
1	*0*	1.2906	0.0185
	1%	1.3417	0.0193
	5%	1.5807	0.0227
	10%	1.9888	0.0286
	15%	2.8052	0.0403
3	*0*	0.9219	0.0120
	1%	0.9301	0.0121
	5%	1.2992	0.0169
	10%	2.7704	0.0360
	20%	2.9061	0.0377
	30%	7.9885	0.1037

Table 4.5: Estimation performance with perturbed assignment matrix.

	Time	RMS Error		RMSN Error	
Scenario	Int.	Estimate	Historical	Estimate	Historical
2	1	2.1727	9.3511	0.0146	0.0629

Table 4.6: Estimation performance in CA/T network; exact-match formulation.

the flow proportions vary, the RMS error jumps to 14.2852.

The second variation of the proportion algorithm uses three targets: sensor counts, deviations from historical flows, and flow proportions. In this case, for the first scenario, the RMS error drops to 9.0282. These results do not look promising in these cases because in this scenario, the historical flows are quite close to the actual flows. If the historical flows were not as reliable, but the historical proportions were reliable, such as in the case of the third scenario, this algorithm would be preferred.

The issue of computational efficiency is important when considering this algo-

rithm. If we consider the historical flow proportions to be very reliable but the historical flows to be not reliable at all, the first variation of the algorithm could be used, which involves the inversion of a matrix of dimension $n_{OD} + n_{\text{prop}}$, where n_{prop} is the number of terms in the proportions target vector. n_{prop} in itself could be much larger than n_{OD}, as described before, depending on which of the three types of objectives is used. The second variant requires the inversion of a matrix of dimension $n_{OD} + n_l + n_{\text{prop}}$ which is even larger. In the case of the Florian network, we have 9 OD pairs, 10 sensors, and 9 terms in the proportions target vector,[2] requiring us to invert a matrix of dimension 28 for the proportion algorithm as opposed to dimension 19 for the standard algorithm, roughly tripling the computation required.

In conclusion, adding OD flow proportions to the estimator can be useful if there is a source of historical calibrated flow proportions. The covariance matrix used in the estimator will be calibrated to indicate the relative confidence in the OD proportions or OD flows themselves. The gain in accuracy realized by the incorporation of additional information in the estimator must be weighed against the computational cost.

4.2.4 Constrained Estimation

The basic estimator for the Irvine network invariably produces an estimate with some negative flows. We have already seen the accuracy of the basic estimator. Using the NNLS algorithm requires a vastly larger number of computations, on the order of 185 *Gflops*, almost three orders of magnitude more than the basic algorithm. This involves many iterations of partitioning the variables and re-estimating. Each

[2]For the first two variations of the proportions objective discussed in section 3.4.1, 9 terms are added in the proportions target: 9 OD pairs, or $3 \times \binom{3}{2} = 9$ pairs of flows. If the third variation were to be used, $\binom{3}{2} = 3$ terms would be added.

re-estimation requires another 500 *Mflop*s, so for this example approximately 370 iterations were performed. The estimate produced has an RMS error of 90.3326, which is only marginally better than the ad hoc version.

It appears from these results that ad hoc methods for ensuring constraints are not violated are preferred to the constrained algorithms. The constrained algorithms produce estimates which are only slightly more accurate than the ad hoc algorithm, while experiencing a severe penalty in computation speed.

4.2.5 Large-Flow Estimator

In the case studies considered here, holding small flows fixed is only applicable to the case of the Irvine network. (The Florian and CA/T networks do not have enough OD pairs to warrant the loss in accuracy that would result.) One task is to determine the threshold of what is considered a small flow. Of the 626 OD pairs in the Irvine network, 350 have flows which are less than 10 vehicles per hour. 456 have flows which are less than 20 vehicles per hour. Using the 10 vehicles per hour cutoff, the estimate has an RMS error of 92.6938 and an RMSN error of 6.5218. This error is better than that of the standard estimator, and the large-flow estimator requires only 64 *Mflop*s, a speedup of a factor of almost 8. These results are summarized in table 4.7.

One caveat about these results: while it appears that the error decreases as we remove more and more variables from the estimator, this is only because we are approaching an estimator which uses only deviations from historical flows as its objective. The RMS error for using the historical database as the estimate is 8.5909. So, the higher the threshold is raised, the closer the estimation error will get to this value. As the representation of the Irvine network is stabilized, and a full calibration is performed, it will be possible to make better judgments as to which algorithms to

Cutoff	Number of Small Flows	RMS Error	RMSN Error	Approx. Speedup
0	*0*	92.6938	6.5218	*1*
10	250	90.3044	6.0865	8
20	456	90.4478	6.0962	24
30	494	88.1943	5.9443	42

Table 4.7: Performance of large-flow estimator on Irvine network.

deploy under what circumstances.

Chapter 5

Summary and Conclusions

This chapter summarizes the main conclusions of this thesis and proposes areas for further research.

The objective of this research was to evaluate alternative formulations and algorithms for real-time OD estimation. Building on the basic OD estimation algorithm and the associated theory, we have:

- developed algorithms to take advantage of additional measurement data and historical databases that might be available
- proposed algorithms with an eye to computational efficiency
- proposed algorithms with nonnegativity and capacity constraints

The various algorithms have been tested on three networks: a simple toy network, Boston's CA/T network, and the ATMS testbed of Irvine, California. The algorithms have been implemented as a component of DynaMIT, the traffic prediction software under development at MIT's ITS laboratory.

We can make some judgments as to which algorithms to deploy under which circumstances. Any algorithm that is tested should be compared to the basic LS

estimator as a base case. In a large, complicated network such as the Irvine network, one or more of the following two possibilities can be implemented: the alternate formulation or holding the small OD flows fixed. In a smaller network, such as the CA/T network, the loss of estimator accuracy is not worth the increase in computation time. As for the constrained algorithms, although their use would be called for in a more complicated network such as Irvine, for which the basic estimate has a higher incidence of negative flows and flows which exceed the capacity constraints, the loss of computational efficiency is too severe. Instead, the ad hoc method of setting the negative flows to zero (or appropriately adjusting the flows downwards) should be used. In a less-complex network, a constrained algorithm would be appropriate.

5.1 Research Directions

There are many areas which should be explored in order to develop a more robust OD estimation model. Several of them will be outlined in this section.

Estimation of Assignment Matrix

Currently the assignment matrix is estimated through an iterative process with OD estimation. However, the estimation of the assignment matrix as it is currently implemented in DynaMIT is quite error-prone. Further research is necessary into real-time estimation methods for the assignment matrix, expanding upon Ashok and Ben-Akiva (2001). In addition, it would be useful to determine under what conditions the iterative process described in section 3.1.3 converges, and to determine an optimal weighting or covariance matrix for the errors in the iterative process.

Placement of Sensors

The discussion in this thesis has assumed that DynaMIT is to be applied to a network where the number of sensors and their locations are fixed. Certainly if there are more OD pairs than sensors, and even possibly if there are more sensors, the estimation performance is at the mercy of the historical database. If the historical database is unreliable, then the estimates will automatically be poor. Also, if it would be desired to use the alternative formulation of matching the counts exactly, it is necessary to remove duplicate sensors or sensors whose counts are multiples of another's from the estimator. With suggestions from the DTA implementer, this pitfall could be avoided. It is important in a planning context when deciding where to install sensors, and how many to install, that the performance of a DTA be taken into account. An important area of research would be to determine the optimum placement of sensors which makes the OD estimation perform the best.

Calibration

While the theoretical model developed in this thesis is sound, much work needs to be done to be able to deploy an accurate version of DynaMIT at a TMC. A comprehensive model for calibrating the DynaMIT input parameters is given in Balakrishna (2001). That study covers all of the important DynaMIT models including route choice parameters, measurement error covariances, transition matrices, and modeling stochasticity in the assignment matrices. In addition, it is necessary to have a good calibration of the supply simulator's speed/density and queuing models.

Appendix A

Kalman Filter Derivation

We derive the Kalman filter solution to the state-space estimation problem[1] of equations 2.4 and 2.5, summarized here:

$$\begin{aligned} y_h &= A_h x_h + v_h \\ x_{h+1} &= \Phi_h x_h + u_h + w_h \end{aligned}$$

where:

- y_h is the measured data,
- x_h is the state of the system,
- v_h and w_h are vectors of random errors,
- u_h is a deterministic input vector,
- A_h, the observation matrix, maps the system state to the measured data, and

[1]This derivation is based on Willsky *et al* (1999).

- Φ_h is the transition matrix, which encapsulates the dynamic behavior of the system.

We assume, for ease of derivation, that v_h and w_h are independent from time interval to time interval, so that $E(v_m v_n') = \mathbf{0}$ for $m \neq n$. Individual vectors v_h and w_h assumed to be zero-mean Gaussian with the measurement noise $v_h \sim \mathcal{N}(\mathbf{0}, C_h)$, and $w_h \sim \mathcal{N}(\mathbf{0}, Q_h)$. The initial state x_0 is assumed to be of a Gaussian distribution (for simplicity), with mean μ_0 and covariance $\Lambda_{0|0} = C_0$.

We would like to estimate the state of the system $x_{h|h} = E(x_h|y_0, y_1, \ldots, y_h)$, the expected state of the system given all of the measurement data. In order to avoid storing all of the previous measurements, we desire a recursive estimator which combines a one-step prediction of the current state of the system with an estimate based on the current measurement data.

As before, we let $\hat{r}_{n|k}$ denote the linear LS estimate of r_n based on all measurements through time period k. The cross-covariance function for a pair of vector processes q and r is indicated by

$$K_{qr}[n,k] = \text{cov}\,(q_n, r_k)$$

which, for zero-mean processes (such as the error processes here) have

$$K_{qr}[n,k] = E(q_n r_k').$$

So for the noise processes v and w we have

$$\begin{aligned} K_{vv}[n,k] &= C_n \delta[n-k] \\ K_{ww}[n,k] &= Q_n \delta[n-k] \\ K_{vw}[n,k] &= K_{wv}[n,k] = \mathbf{0} \end{aligned}$$

where

$$\delta[n-k] = \begin{cases} 0 & n \neq k \\ 1 & n = k \end{cases}$$

We let $\Lambda_{n|k}$ denote the covariance of the error

$$e_{n|k} = x_n - \hat{x}_{n|k}$$

based on measurements through time period k, so that

$$\Lambda_{n|k} = E\left(e_{n|k}e'_{n|k}\right).$$

One important result from estimation theory is the orthogonality principle:

Theorem 1 (Orthogonality) *A linear estimator $\hat{x}_L(\cdot)$ is the linear least-squares estimator if and only if the associated estimation error $e(x,y) = \hat{x}_L(y) - x$ is orthogonal to* any *vector-valued linear function of the data; that is*

$$E\left[(\hat{x}_L(y) - x)(Fy + g)'\right] = \mathbf{0}$$

for any constant matrix F and any constant vector g.

From this theorem we have that the filtered estimates $\hat{x}_{n|n}$ satisfy

$$E\left[\left(\hat{x}_{n|n} - x_n\right) y'_k\right] = \mathbf{0}$$

for all $k \leq n$ and the predictions also satisfy

$$E\left[\left(\hat{x}_{n|n-1} - x_n\right) y'_k\right] = \mathbf{0}$$

We define the *innovation*

$$z_n = y_n - \hat{y}_{n|n-1}$$

as the part of the current measurement that is uncorrelated with the previous measurement samples.[2] The innovations process is white, (each innovation is uncorrelated

[2]This notion of an "estimate" of the next measurement is used to express the idea that measurements are correlated from time interval to time interval. Then, there is a part of the measurement that we can predict from the previous data, indicated by $\hat{y}_{n|n-1}$ and a part, the innovation, (z_n) that is uncorrelated with the previous data, and that we cannot predict.

with the previous ones) so that

$$K_{zz}[n,k] = \Lambda_z \delta[n-k].$$

Willsky *et al* (1999) demonstrate that the estimate $\hat{x}_{n|n}$ based on the observations of the innovations, can be written as the output of a linear filter, or

$$\hat{x}_{n|n} = \sum_{k=0}^{n} h[n,k] z_k.$$

This results in the *Wiener-Hopf* equation, which states that the cross-covariance of the estimate and innovations must satisfy

$$K_{xz}[n,k] = \sum_{s=0}^{n} h[n,s] K_{zz}[s,k]$$

Rearranging this equation using techniques of Fourier analysis gives what is known as the *Wiener filter* for the estimate $\hat{x}_{n|n}$. Realizing that the deterministic input sequence u also contributes to the estimate, we can write

$$\begin{aligned} \hat{x}_{n|n} &= \sum_{k=0}^{n} K_{xz}[n,k] K_{zz}^{-1}[k,k](z_k - u_k) \\ &= \hat{x}_{n|n-1} + K_n(z_n - u_n) \end{aligned} \tag{A.1}$$

where

$$K_n = K_{xz}[n,n] K_{zz}^{-1}[n,n] \tag{A.2}$$

This is the form of the "corrected" estimator, which is the sum of the "predicted" estimate $\hat{x}_{n|n-1}$ with a correction term based on the new measurements.

Also, the predicted estimate, before the new measurement data is received, can be written

$$\hat{x}_{n|n-1} = u_{n-1} + \sum_{k=0}^{n-1} K_{xz}[n,k] K_{zz}^{-1}[k,k] z_k$$

The filtered error covariance can be written

$$\begin{aligned}\Lambda_{n|n} &= \Lambda_{x,n|n} - \sum_{k=0}^{n} K_{xz}[n,k]K_{zz}^{-1}[k,k]K'_{xz}[n,k] \\ &= \Lambda_{e,n|n-1} - K_n K'_{xz}[n,n] \end{aligned} \tag{A.3}$$

since

$$\Lambda_{n|n-1} = \Lambda_{x,n} - \sum_{k=0}^{n-1} K_{xz}[n,k]K_{zz}^{-1}[k,k]K'_{xz}[n,k]$$

The transition equation gives us the estimate

$$\hat{x}_{n+1|n} = \Phi_n \hat{x}_{n|n} + u_n. \tag{A.4}$$

Then the error for this estimate is

$$\begin{aligned} e_{n+1|n} &= x_{n+1} - \hat{x}_{n+1|n} \\ &= (\Phi_n x_n + w_n + u_n) - (\Phi_n \hat{x}_{n|n} + u_n) \\ &= \Phi_n(x_n - \hat{x}_{n|n}) + w_n \\ &= \Phi_n e_{n|n} + w_n, \end{aligned}$$

a nice simplification, since the error follows the same dynamic process (with the absence of the deterministic input u_n) that the state does. So then

$$\begin{aligned}\Lambda_{n+1|n} &= E[e_{n+1|n}e'_{n+1|n}] \\ &= \Phi_n E[e_{n|n}e'_{n|n}]\Phi'_n + E[\Phi_n e_{n|n}w'_n] + E[w_n e'_{n|n}\Phi'_n] + E[w_n w'_n] \\ &= \Phi_n \Lambda_{n|n}\Phi'_n + Q_n \end{aligned} \tag{A.5}$$

since the state estimation error is uncorrelated with the noise process, and so the cross terms drop out.

Using the measurement equation, and the fact that the measurement noise process is uncorrelated with the prior measurements, we have

$$\hat{y}_{n|n-1} = A_n \hat{x}_{n|n-1}.$$

Using the definition of the innovation from above, we have

$$\begin{aligned} z_n &= y_n - A_n\hat{x}_{n|n-1} \\ &= y_n - A_n(x_n - e_{n|n-1}) \\ &= A_n e_{n|n-1} + v_n \end{aligned} \tag{A.6}$$

since $y_n = A_n x_n + v_n$. We can use this expression for the innovation to derive that cross-covariances

$$\begin{aligned} K_{xz}[n,n] &= E[x_n z_n'] \\ &= E[x_n e_{n|n-1}']A_n' + E[x_n v_n] \\ &= E[(\hat{x}_{n|n-1} + e_{n|n-1})e_{n|n-1}']A_n' + \mathbf{0} \\ &= \Lambda_{n|n-1}A_n' \end{aligned} \tag{A.7}$$

by the orthogonality principle. Also,

$$\begin{aligned} K_{zz}[n,n] &= E[z_n z_n'] \\ &= A_n E[e_{n|n-1}e_{n|n-1}']A_n' + \text{cross terms} + E[v_n v_n'] \\ &= A_n\Lambda_{n|n-1}A_n' + C_n \end{aligned}$$

and so the Kalman gain matrix as defined above in equation A.2 becomes

$$K_n = \Lambda_{n|n-1}A_n'\left(A_n\Lambda_{n|n-1}A_n' + C_n\right)^{-1} \tag{A.8}$$

Using the above derivation we can produce the Kalman filter algorithm. The "predicted" estimate, based solely on the prior state and measurements, can be written using equation A.4, and its associated covariance using equation A.5. The Kalman gain matrix was given just above. The "corrected" estimate, based on the new measurements as well as the prior state and measurements, can be written from equation

A.1 and the definition of the innovation, and its associated covariance is found in equation A.3 using equation A.7. In summary, for each iteration of the filter, we have the following four steps:

1. Generate the next estimate and its associated error covariance. These are the predictor equations, using information only from the state transition equation.

$$\begin{aligned} \hat{x}_{h|h-1} &= \Phi_{h-1}\hat{x}_{h-1|h-1} + u_{h-1} \\ \Lambda_{h|h-1} &= \Phi_{h-1}\Lambda_{h-1|h-1}\Phi'_{h-1} + Q_{h-1} \end{aligned}$$

2. Compute the Kalman gain matrix:

$$K_h = \Lambda_{h|h-1}A'_h(A_h\Lambda_{h|h-1}A'_h + C_h)^{-1}$$

3. Generate the filtered estimate and its associated error covariance. These are the corrector equations, using information only from the measurement equation.

$$\begin{aligned} \hat{x}_{h|h} &= \hat{x}_{h|h-1} + K_h(y_h - u_h - A_h\hat{x}_{h|h-1}) \\ \Lambda_{h|h} &= \Lambda_{h|h-1} - K_hA_h\Lambda_{h|h-1} \end{aligned}$$

4. Increment h and go back to step 1.

Appendix B

Constrained Least Squares Algorithms

B.1 Nonnegative Least Squares

The following is the nonnegative least squares algorithm as presented by Lawson and Hanson (1974) to solve the problem

$$\min ||Ax - y|| \quad \text{subject to} \quad x \geq 0$$

with $A \in \mathbb{R}^{m \times n}$ and $y \in \mathbb{R}^m$. $\mathcal{P}$ is the index set of variables of the estimate $x \in \mathbb{R}^n$ which are positive, and $\mathcal{Z}$ is the index set of variables which are set to 0.

The algorithm consists of the following steps:

1 Set $\mathcal{P} = \emptyset$, $\mathcal{Z} = \{1, 2, \ldots, n\}$, and $x = 0$.

2 Compute the dual vector $w = A'(y - Ax)$.

3 If the set $\mathcal{Z}$ is empty or if $w(j) \leq 0$ for all $j \in \mathcal{Z}$, go to step 13.

4 Find the index $t \in \mathcal{Z}$ such that $w(t) = \max_{j \in \mathcal{Z}} w(j)$.

5 Move the index t from set $\mathcal{Z}$ to set $\mathcal{P}$.

6 Let $A_{\mathcal{P}}$ be the $m \times n$ matrix which is the same as A except that the columns of $A_{\mathcal{P}}$ which are indexed in $\mathcal{Z}$ are set to columns of zeros.

7 Compute z as a solution of the LS problem $A_{\mathcal{P}}z \approx y$. Let $z(j) = 0$ for $j \in \mathcal{Z}$.

8 If $z(j) > 0$ for all $j \in \mathcal{P}$, set $x = z$ and go to step 2.

9 Find an index $q \in \mathcal{P}$ such that $x(q)/(x(q) - z(q))$ is minimized, with $z(q) \leq 0$.

10 Set $\alpha = x(q)/(x(q) - z(q))$.

11 Set $x = x + \alpha(z - x)$.

12 Move from set $\mathcal{P}$ to set $\mathcal{Z}$ all indices j for which $x(j) = 0$. Go to step 6.

13 The computation in complete. x is the solution.

B.2 Least Distance Programming for the LSI Problem

Lawson and Hanson also give an algorithm for solving the LDP problem

$$\min ||z|| \quad \text{subject to} \quad Cx \geq b$$

The algorithm can detect when the inequality constraints are inconsistent; that is, when there is no solution space. The algorithm proceeds as follows:

1 Define the $(n+1) \times m$ matrix $E = [C \quad b]'$ and the vector $f = [0 \ldots 01]'$ of length $n + 1$.

2 Use the NNLS algorithm to compute a vector $\hat{u} \in \mathbb{R}^m$ to solve $\min ||Eu - f||$ subject to $u \geq 0$.

3 Compute $r = E\hat{u} - f$.

4 If $||r|| = 0$ then the constraints are inconsistent, and the computation terminates.

5 Compute the optimal solution $\hat{x}$ with $\hat{x}(j) = -r(j)/r(n + 1)$.

Bibliography

[1] *City of Irvine annual traffic management report.* Irvine, California, Department of Public Works and Department of Public Safety, 1999-2000.

[2] B. ABDULHAI, *California's ATMIS testbed: Expanding the frontiers of traffic management*, Intellimotion, **7** (1998), no. 1, pp. 1, 8–9.

[3] C. ANTONIOU, *Demand simulation for dynamic traffic assignment*, Master's thesis, Massachusetts Institute of Technology, 1997.

[4] K. ASHOK, *Dynamic trip table estimation for real time traffic management systems*, Master's thesis, Massachusetts Institute of Technology, 1992.

[5] ——, *Estimation and Prediction of Time-Dependent Origin-Destination Flows*, PhD thesis, Massachusetts Institute of Technology, 1996.

[6] K. ASHOK AND M. E. BEN-AKIVA, *Dynamic origin-destination matrix estimation and prediction for real-time traffic management systems*, in International Symposium on Transportation and Traffic Theory, C. F. Daganzo, ed., Elsevier Science, 1993, pp. 465–484.

[7] ——, *Estimation and predcition of dynamic origin-destination flows by combining different types of measurements*, tech. rep., Massachusetts Institute of Technology, 1997.

[8] ——, *Alternative approaches for real-time estimation and prediction of time-dependent origin-destination flows*, Transportation Science, **34** (2000), no. 1, pp. 21–36.

[9] ——, *Estimation and prediction of time-dependent origin-destination flows with a stochastic mapping of path flows and link flows*, Transportation Science, **35** (2001). Forthcoming.

[10] R. BALAKRISHNA, *Calibrating the OD estimation / prediction and route choice models in DynaMIT*, Master's thesis, Massachusetts Institute of Technology, 2001.

[11] M. E. BEN-AKIVA, *Methods to combine different data sources and estimate origin-destination matrices*, in Gartner and Wilson [26], pp. 459–481.

[12] M. E. BEN-AKIVA, M. BIERLAIRE, D. BURTON, H. N. KOUTSOPOULOS, AND R. MISHALANI, *Network state estimation and prediction for real-time transportation management applications.* Forthcoming, 2001.

[13] D. BERTSIMAS AND J. N. TSITSIKLIS, *Introduction to Linear Optimization*, Athena Scientific, Belmont, MA, 1997.

[14] Å. BJÖRCK, *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics, Philadelphia, 1996.

[15] J. BOTTOM, *Development of ITS analysis tools for Lower Westchester County.* Proposal to New York State Department of Transportation, July 1999.

[16] G. E. CANTARELLA AND E. CASCETTA, *Dynamic processes and equilibrium in transportation networks: Towards a unifying theory*, Transportation Science, **29** (1995), pp. 305–329.

[17] E. CASCETTA, *Estimation of trip matrices from traffic counts and survey data: A generalized least squares estimator*, Transportation Research B, **18** (1984), no. 4/5, pp. 289–299.

[18] E. CASCETTA, D. INAUDI, AND G. MARQUIS, *Dynamic estimators of origin-destination matrices using traffic counts*, Transportation Science, **27** (1993), no. 4, pp. 363–373.

[19] E. CASCETTA AND M. N. POSTORINO, *Fixed point approaches to the estimation of O/D matrices using traffic counts on congested networks*, Transportation Science, **35** (2001), no. 2. Forthcoming.

[20] A. CHACHICH. Personal communication, May 2001.

[21] C. K. CHUI AND G. CHEN, *Kalman Filtering with Real-Time Applications*, no. 17 in Springer Series in Information Sciences, Springer-Verlag, Berlin, 2 ed., 1991.

[22] T. COLEMAN, M. A. BRANCH, AND A. GRACE, *Optimization Toolbox User's Guide for use with MATLAB*, The MathWorks, Natick, MA, 1999.

[23] R. DEUTSCH, *Estimation Theory*, Prentice-Hall, Englewood Cliffs, NJ, 1965.

[24] E. R. DOUGHERTY, *Probability and Statistics for the Engineering, Computing, and Physical Sciences*, Prentice-Hall, Englewood Cliffs, NJ, 1990.

[25] L. A. ESCOBAR AND B. SKARPNESS, *A closed form solution for the least squares regression problem with linear inequality constraints*, Communications in Statistics: Theory and Methods, **13** (1984), no. 9, pp. 1127–1134.

[26] N. H. GARTNER AND N. H. M. WILSON, eds., *Proceedings of the Tenth International Symposium on Transportation and Traffic Theory*, Cambridge, MA, July 1987, Elsevier Science.

[27] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 1996.

[28] R. J. HANSON AND K. H. HASKELL, *Two algorithms for the linearly constrained least squares problem*, ACM Transactions on Mathematical Software, **8** (1982), no. 3, pp. 323–333.

[29] S. M. KAY, *Fundamentals of Statistical Signal Processing: Estimation Theory*, Prentice-Hall PTR, Englewood Cliffs, NJ, 1993.

[30] C. L. LAWSON AND R. J. HANSON, *Solving Least Squares Problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[31] T. O. LEWIS AND P. L. ODELL, *Estimation in Linear Models*, Prentice-Hall, Englewood Cliffs, NJ, 1971.

[32] C. K. LIEW, *Inequality constrained least-squares estimation*, Journal of the American Statistical Association, **71** (1976), no. 355, pp. 746–751.

[33] J. M. MENDEL, *Lessons in Digital Estimation Theory*, Prentice-Hall, Englewood Cliffs, NJ, 1987.

[34] E. NIVER, K. C. MOUSKOS, T. BATZ, AND P. DWYER, *Evaluation of the TRANSCOM's system for managing incidents and traffic (TRANSMIT)*, IEEE Transactions on Intelligent Transportation Systems, **1** (2000), no. 1, pp. 15–31.

[35] I. OKUTANI, *The Kalman filtering approaches in some transportation and traffic problems*, in Gartner and Wilson [26], pp. 397–416.

[36] W. H. PRESS, B. P. FLANNERY, S. A. TEUKOLSKY, AND W. T. VETTERLING, *Numerical Recipies: The Art of Scientific Computing*, Cambridge University Press, Cambridge, 1989.

[37] P. SAMUEL, *Dash from cash*, ITS International, (January / February 1999), no. 20, pp. 40–46.

[38] H. D. SHERALI AND T. PARK, *Estimation of dynamic origin-destination trip tables for a general network*, Transportation Research B, **35** (2001), pp. 217–235.

[39] R. T. STEFANI, C. J. SAVANT, B. SHAHIAN, AND G. H. HOSTETTER, *Design of Feedback Control Systems*, Oxford University Press, 1994.

[40] J. STOER, *On the numerical solution of constrained least-squares problems*, SIAM Journal on Numerical Analysis, **8** (1971), no. 2, pp. 382–411.

[41] G. STRANG, *Linear Algebra and its Applications*, Harcourt Brace Jovanovich, Fort Worth, TX, 1988.

[42] C. SUN, *Dynamic origin/destination estimation using true section densities*, tech. rep., California Partners for Advanced Transit and Highways, 2000.

[43] N. J. VAN DER ZIJPP, *Dynamic Origin-Destination Matrix Estimation on Motorway Networks*, PhD dissertation, Delft University of Technology, Faculty of Civil Engineering, May 1996.

[44] ——, *Dynamic OD-matrix estimation from traffic counts and automated vehicle identification data*, Transportation Research Record, (1997), no. 1607.

[45] H. J. WERNER, *On inequality constrained generalized least-squares estimation*, Linear Algebra and its Applications, **127** (1990), pp. 379–392.

[46] A. S. WILLSKY, G. W. WORNELL, AND J. H. SHAPIRO, *Stochastic processes, detection and estimation.* Notes for MIT course 6.432, 1999.

[47] Q. YANG, *A Simulation Laboratory for Evaluation of Dynamic Traffic Management Systems*, PhD thesis, Massachusetts Institute of Technology, 1997.

[48] Q. YANG AND H. N. KOUTSOPOULOS, *A microscopic traffic simulator for evaluation of dynamic traffic management systems*, Transportation Research C, **43** (1996), pp. 113–119.