Network Knowledge and Route Choice

by

Michael Scott Ramming

B.S., Civil Engineering, Washington University in St. Louis, May 1991

M.S., Transportation, Massachusetts Institute of Technology, May 1994

Submitted to the Department of Civil and Environmental Engineering in
Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy in Transportation

at the

Massachusetts Institute of Technology

February 2002

Signature of Author____
Department of Civil and Environmental Engineering
28 December 2001

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

Accepted by____
Oral Buyukozturk
Chairman, Departmental Committee on Graduate Studies

Network Knowledge and Route Choice

by Michael Scott Ramming

Submitted to the Department of Civil and Environmental Engineering
on 28 December 2001 in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Transportation

Abstract

Models of urban traveler route choice are reviewed in the context of Intelligent Transportation Systems, particularly Advanced Traveler Information Systems. Existing models suffer from assumptions of perfect information about travel conditions and infinite information processing capabilities of drivers. We present evidence that a majority of travelers fail to minimize travel time or distance. We also show that travelers with more network knowledge appear to vary their commute route to respond to changing travel conditions. Coefficient estimates of a model of network knowledge, based on the geographical idea of spatial ability, are presented. To better understand habitual route choice behavior, we examine many possible route generation algorithms. A simulation approach is preferred because it allows for heterogeneity in driver perceptions and it has a quick computational time. Alternative route choice model specifications such as Multinomial Logit, C-Logit, Path Size Logit, Cross-Nested Logit and Logit Kernel Probit are evaluated. The exponential specification of the Path Size term, using a large parameter value, offers a considerable improvement in fit over MNL, C-Logit and CNL. A hybrid Path Size Logit and Logit Kernel Probit model offers the best overall fit; however, the stability of these estimates requires further examination. The hybrid Path Size Logit and CNL model provides the next best empirical fit. Random coefficient specifications of MNL, PSL and LK Probit models were also examined. Significant random coefficient parameter estimates were only obtained for the MNL model. This result suggests that random coefficients capture variation in route choice models that would be more effectively explained by a Path Size or LK Probit specification. Model fit can be further improved by adding an Implicit Availability/Perception term that includes estimated network knowledge. However, this term provides limited explanatory power, as can be seen by its standard errors and by forecasts that are relatively insensitive to changes in traveler knowledge. These results suggest that continued development of better attitudinal surveys to assess network knowledge and wayfinding strategies would allow estimation of route choice models with better explanatory power.

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

Acknowledgements

Practically all university catalogs mention that Ph.D. dissertations must represent an original contribution to research. Practically all dissertation acknowledgements state that the contents are not so much an individual's contribution but the product of many individuals offering their insights and mentoring to the author. In that regard, this dissertation is no different. Among those deserving thanks are:

Moshe Ben-Akiva, my adviser and committee chair, who provided considerable insight into the discrete choice models used in this thesis; who had the difficult task of encouraging me to work independently and efficiently; and who introduced me to so many helpful colleagues.

My thesis committee members: Joe Sussman, who kept insisting that we examine the big picture question of what do we want to accomplish with this thesis. Nigel Wilson, who offered many suggestions for the technical content of this thesis, and who reminded me of the importance of balancing my research and non-research life to avoid stress. Shlomo Bekhor, who offered much encouragement in the last few months. Kalidas Ashok, Michel Bierlaire, Mithilesh Jha, Satosi Yamasita, Eiji Hato, and Ikki Kim, who attended committee meetings out of a common research interest and offered many helpful suggestions.

Denis Bolduc, who provided customized code to perform the Logit Kernel route choice estimation described in this thesis. Galit Toledo, my colleague on the Lexington, KY, route choice modeling and wayfinding survey projects. UROPs Divya Agarwalla and Andrew Mamo, who assisted with TransCAD tasks for this thesis.

Funding sources, including the Federal Highway Administration's Eisenhower Fellowship; a University Transportation Center research grant; and NSF Award Number 9986475, Individuals' Spatial Abilities and Wayfinding in Transportation Networks. I am also grateful to Charles River Associates, the MIT Planning Office, and MADDIF project led by André de Palma and Cédric Fontan for providing consulting opportunities. Finally, I must thank my parents and the federal Perkins Program for support of my living expenses.

I'm also blessed that my family – particularly Mom and Dad – took turns calling on Sunday nights to listen to my latest snag and to help establish a rhythm for beginning the work week.

I'm thankful for "Slowpoke Club" of Todd Olmstead, Jon Bottom, Joan Walker and myself. We met regularly for awhile to report and progress and encourage each other to keep up the pace of our research. I'm particularly thankful to Joan for all her help with TransCAD. I'm also glad for the opportunity to get to know many officemates, including Kazi Ahmed, John Bowman, Sarah Bush, Amber Crabbe, Xiaojing Dong, Jenny Farver, Song Gao, Andrew Glenn, Nathaniel Grier, Taka Hosoda, Shan Lan, Seong-Cheol Kang, Lisa Klein, Matthew Kurian, Amalia Polydoropoulou, Sreeram Thirukkonda, and Eric Wile.

Leanne Russell-Simpson, Julie Bernardi and Cheryl Thornton-Gillespie all served as administrative assistants to my adviser during my studies, which means that they were among the people I often counted on for help with research presentations. I'm also grateful for the support of Cynthia Steward and Sydney Miller, graduate administrators for Civil and Environmental Engineering, and the Center for Transportation Studies, respectively.

My friends Todd Stefanik, Matt Bruce, Denise Brehm and Dinesh Gopinath, and the Lutheran-Episcopal Ministry were all great listeners when things got hairy. I also appreciate the opportunity to learn to play hockey at MIT, and for my teammates on Slush, Frost Heaves, Chemistry LN2, Loonies, Can-Am Gladiators, Speds, and South Shore Rebels, and at practice with the Lobsters. Likewise, I'm glad I got the chance to meet great people working with the Graduate Student Council and the Association of Student Activities, including Van Chu, Rick Gresh, Brian Lee, Maria Raposo and Matt McGann.

I'm thankful for the many people who served as resources to help me work more efficiently and take better care of myself. These people include Elisa Birdseye, Mariana Castells, Debby Clendaniel, Christine Girard, Richard Goldhammer, David Reniere, Lorraine Toher, Alan Tweedy and Michael Wechsler.

During my studies at MIT, I have been blessed by hearing and meeting many prophets. These people include Fred Salvucci, Jonathan Richmond, Hank Dittmar, Matthew of common cathedral, Harry of common cathedral, Christopher Allen-Doucot, Desmond Tutu, and Steven Charleston.

Finally, I give thanks to the God who made travelers, hockey sticks and other random stuff; and who inspired Marianne Williamson (1992) to write the following poem – often misattributed to Nelson Mandela's 1994 inauguration speech – which I kept a copy taped to the wall in my office:

Our deepest fear is not that we are inadequate.
Our deepest fear is that we are powerful beyond measure.
It is our light, not our darkness that most frightens us.
We ask ourselves,
Who am I to be brilliant, gorgeous, talented, fabulous?
Actually, who are you not to be?
You are a child of God.
Your playing small does not serve the world.
There is nothing enlightened about shrinking so that other people won't feel insecure around you. We are all meant to shine, as children do.
We were born to make manifest the glory of God that is within us.
It is not just in some of us; it is in everyone.
And as we let our own light shine, we unconsciously give other people permission to do the same.
As we are liberated from our own fear, our presence automatically liberates others.

Contents

Appendices

List of Figures

List of Tables

1
Introduction

Urban residents must travel from place to place to earn a living, and to acquire the goods and services necessary to life and that make life more enjoyable. Transportation planners therefore have a challenging task of predicting and trying to improve travel flows, which depend on many complex human factors including land use, preferences and perceptions of the built environment.

This thesis considers the question of how awareness of alternative routes affects travelers' choices. Important aspects of this question involve how a traveler's awareness of alternative routes can be predicted, and since travelers can only choose routes of which they are aware, how the chosen route can be predicted while considering the traveler's awareness. The approach presented here should be familiar to travel demand modelers. Discrete choice models and random utility theory provide the basic framework for describing awareness and choice. We introduce the concept of network knowledge to describe unseen differences among travelers, which influences their travel patterns. Network knowledge is a latent variable that influences both the set of alternatives available to a traveler (often called the traveler's *choice set* or *consideration set*), and his or her utility. Socioeconomic characteristics and attitudinal responses are used to infer a traveler's underlying network knowledge.

The idea of network knowledge and ability is inspired by psychologists' more narrow definition of spatial ability. In the same way that IQ tries to measure a person's ability to learn rather than the content of his or her knowledge, spatial ability refers to a person's mental capability to learn, organize and recall spatial information. This thesis uses the term network ability to indicate the capability to acquire and process information about transportation networks. The content of that stored information may be called spatial knowledge, network knowledge or the choice set.

This thesis does not attempt to describe the biological or mental processes by which network ability might arise, or even how travelers may gain knowledge or awareness of particular routes. Nor is this thesis meant to provide a short term description of traveler behavior, such as en-route diversion in response to observed congestion ahead. Instead, a longer-term modeling horizon is considered. The objective is to predict the considered and chosen routes of a traveler given his or her characteristics. To do this, we consider the likely level of spatial ability he or she possesses, and given that spatial ability, estimated the choice probabilities of the considered routes. Models with this type of structure are applicable for questions of how travelers may change their patterns when new transportation facilities and services become available, and how information services may affect travelers decisions by making them aware of a greater number of alternatives.

1.1 Problem Motivation

This research topic is motivated by Advanced Traveler Information Systems (ATIS), which are one component of Intelligent Transportation Systems (ITS). Unlike conventional transportation infrastructure building projects, which attempt to improve transportation network performance by providing increased physical capacity, ITS seeks to use information technology to better operate the existing transportation infrastructure. ATIS is expected to improve travel flows and mitigate congestion by alerting travelers of current conditions throughout the network, and allowing them to avoid congested routes. Alternatively, an ATIS may recommend an optimal route to the traveler. The effectiveness of such an ATIS would therefore depend on driver compliance (that is, adoption of the suggested route), and the quality of the information from which the guidance is generated.

However, the motivation for ATIS is inconsistent with the assumptions of conventional travel demand models. Conventional models assume travelers have full information about the network (or the same information that is available to the analyst), and choose the best route from among all those available. Stochastic user equilibrium models are often applied by using these assumptions and the further assumption that travelers have uniform perception errors. Instead, ATIS proponents suggest that travelers do not have full knowledge of the transportation network, or at least not full knowledge of current travel conditions. The various sensing, processing and dissemination components of ATIS provide more realistic descriptions of current travel conditions. However, it is realistic to expect that no sensor can be perfectly accurate, and no dissemination system can be instantaneous. Instead, a more realistic set of assumptions for travel demand modeling is that travelers have imperfect and incomplete knowledge of the transportation network, and that ATIS provides less fallible information to travelers.

This research project attempts to describe how travelers behave in the setting of incomplete and imperfect information. Many simplifying assumptions will have to be made to address this problem. For example, this thesis considers the static case – what routes are travelers aware of, under normal conditions – and ignores the benefit of providing information about non-recurrent congestion. Therefore, this study is better able to address the question of “how do travelers make use of conventional information?” than “how will travelers make use of ATIS information?” Often, a complicated question may highlight the fact that a supposedly simple question must be answered first.

This study also focuses on automotive route choice, rather than public transportation or multi-modal route choice. The operation of public transportation as vehicles at specific headways, stopping at selected locations and with set fare structures, adds additional dimensions to awareness which are beyond the scope of current research.

1.2 Contributions

This research effort attempts to make several contributions to travel demand modeling, including

- developing a framework for incorporating network knowledge in the route choice context, and identifying relationships between readily-available socioeconomic variables, network knowledge and travel choices;
- developing and evaluating choice set generation procedures for route choice that produce realistic alternative routes; and
- developing the Path-Size Logit and the Logit Kernel Probit models for route choice; estimating them and other well-known route choice model specifications using survey data and a large urban network; and comparing the estimation results and model forecasts of each model type.

Each of these contributions is described in the following sections.

1.2.1 Identify Relationships with Network Knowledge

This study takes the unique approach of using geographers' concept of spatial knowledge as an explanatory variable in a route choice model system.

Clearly, we can no longer assume travelers know the entire transportation system in the metropolitan area and simply choose the least-distance or least-time path. We assert that travelers' network knowledge affects the routes available from which a traveler may choose, and we wish to relate this to some readily-observable socio-economic or attitudinal variables. Therefore, we develop several techniques to incorporate the influence of network knowledge in transportation choice models. Some of these techniques explicitly estimate a quantity that may be interpreted as a value of network knowledge, while in other techniques, the role of network knowledge is more implicit.

First, we present various attitudinal questions that may be used to assess survey respondents' levels of knowledge of transportation facilities and their wayfinding strategies. Attitudinal questions have been used in choice surveys to assess other traveler preferences, such as the monetary value of time, or the importance of reliability or amenities. The work presented in this thesis represents one of the first cooperative efforts between transportation professionals and geographers to develop a survey instrument for assessing network and wayfinding abilities.

Also, we examine different link attributes that may be associated with the idea of *prominence* – that some facilities have special characteristics that make them more likely to be known by a wider number of travelers. In particular, prominent facilities are those that unfamiliar and novice travelers would learn first. We examine model specifications with interactions among network knowledge – a traveler characteristic – and prominence attributes of alternative routes. These specifications include pure multiplicative interaction terms in a Multinomial Logit (MNL) context, an interaction as an IAP Logit term, and a more elaborate binary logistic IAP Logit term.

We examine alternative formulations of the concept of Path Size to empirically establish which formulation best represents travelers' perceptions of overlapping paths. The Path Size term, which is discussed in the next section, imbeds travelers' perceptions of alternative paths in a measure of the "significance" or "relevance" of a path relative to others in the choice set. We also test the Path-Size (PS) Logit formulation against other model types that represent path overlapping – C-Logit, Cross-Nested Logit and Logit Kernel. These model types are discussed in Chapter 2.

1.2.2 Validate Choice Set Generation Procedures for Route Choice

During the process of developing a model of route choice, we must first construct possible alternative routes that travelers may consider. Ideally, these alternative routes should be generated by a set of objective algorithms, so that the techniques presented in this thesis may be replicated by practitioners. Stopher (1980) and Williams and Ortuzar (1982) have shown empirically that considerable biases (more precisely, statistical inconsistency) in utility parameter estimates result when choice sets are poorly or capriciously constructed. Swait and Ben-Akiva (1986) present a proof of the bias that results from a mis-specified choice set. A set of objective path generation algorithms will therefore help avoid such biases. Of course, for the route choice model to be estimable, the chosen alternative must appear in the choice set, and to maintain objectivity, the path generation algorithm should be able to replicate the routes travelers report using. A procedure of generating several routes according to some objective criteria, and then adding the chosen route if it wasn't included among the generated routes may be attractive, and perhaps even useful as an intermediate step to better understanding traveler choices. However, this procedure lacks the objectivity necessary for statistical modeling. A path generation procedure that does not generate the chosen path is an indication that something is lacking in the path generation procedure. Path generation algorithms capable of reflecting human choices would be useful for more purposes than just estimating models of traveler behavior, as described in the next paragraph.

ATIS developers want their products and services to be useful for their customers. However, an early in-vehicle route guidance test in the Chicago metropolitan area by Schofer, Koppelman and Charlton (1997) revealed considerable consumer dissatisfaction with the routes generated by algorithms similar to those used in traditional transportation planning models (e.g., shortest travel time). That is, if travelers choose their routes based on travel time and many other considerations, such as minimizing the difficulty of the driving task or maximizing facility continuity, but an ATIS suggests routes based on travel time only, the ATIS will not suggest routes that travelers would consider the most attractive. Instead, algorithms that generate more "human-looking" routes would have greater usefulness and appeal to ATIS customers.

Once reasonable feasible routes have been identified, we can turn to the task of determining which routes a traveler considers when making her or his choice. This step is called "choice set generation" in discrete choice literature because in this step the alternatives in a traveler's choice set are enumerated. The predicted choice probability is a function of the attributes of all the alternatives in an individual's choice set.

Ben-Akiva and Boccarà (1995) have estimated a probabilistic choice set generation model for mode choice in Maceio, Brazil. However, in the mode choice setting, the number of possible alternatives is small (e.g., auto driver, auto passenger, bus and taxi), while in the automotive choice context, the number of possible routes can be rather large. The large number of alternatives, and the fact that routes may overlap, considerably complicates choice set generation modeling.

Cascetta and Papola (1998) have proposed the Implicit Availability/Perception (IAP) Logit model to reduce the computational complexity of choice set generation for route choice. Instead of explicitly predicting the alternatives available to, or considered by, a traveler, they include a correction term that implicitly accounts for the availability of an alternative. The correction term is a logarithmic transform of a variable taking values between zero and one. When the alternative is available, this variable is one, the correction term is zero and thus has no impact on utility. In the other case, as the variable approaches zero, the correction term and thus the alternative's utility approaches negative infinity. The alternative then makes no contribution in the denominator of a logit model. Intermediate values of this zero-one variable correspond to situations in which the analyst does not know whether the alternative is available.

This thesis develops the techniques to consider travelers' choice sets explicitly. Examining the effect of various factors on awareness of routes has useful implications for practitioners. Not only can the impact of information systems be evaluated, but if the variables affecting awareness can be influenced by transportation policies, then perhaps more effective congestion reduction programs can be developed.

1.2.3 Estimate and Compare Route Choice Models

Finally, this project makes a contribution by overcoming a particular challenge of logit route choice models. Path overlap is often cited as a limitation of multinomial logit. We present the Path-Size Logit model, which includes an explicit correction for overlapping routes. Logit size corrections have been used for aggregate alternatives common in destination choice settings, including workplace location and residence choice. The results presented in this thesis represent the first application of size corrections for route choice.

The PS Logit model has several advantages over other methods. Ignoring the path overlap problem is incorrect. The STOCH algorithm, which results in a multinomial logit split between "reasonable" or "efficient" paths (those containing only links that take the traveler farther from the origin and closer to the destination), also suffers from this limitation of MNL. The C-Logit model, proposed by Cascetta, Nuzzolo, Russo and Vitetta (1996), offers several different formulations for adjusting for path overlap; however, the authors offer no guidance or theoretical basis for the selection of one functional form over another. The Cross-Nested Logit model, first developed by Vovsha and Bekhor (1998), is theoretically attractive, but difficult to implement.

The Logit Kernel (LK) Probit model of Ben-Akiva and Bolduc (1996) allows for the flexible error covariance structure associated with the Probit model, and the simplicity of MNL should the Probit error terms prove insignificant. Because of the Probit error terms, LK Probit in computationally demanding to estimate, but offers a general structure that may better explain the choice data.

By estimating each model from the same data set, we have a consistent basis from which to make observations and recommendations relating to computational times and explanatory power. We also estimate hybrid models – for example, C-Logit with Path Size, CNL with IAP, or Logit Kernel with Path Size – to examine the explanatory power of the various specifications. We also consider some sample enumeration applications, to confirm the models produce reasonable forecasts.

1.3 Outline of the Thesis

Chapter 2 summarizes the relevant literature regarding route choice, travel information, wayfinding and cognition. Chapter 3 describes the behavioral hypotheses that underlie this effort, and presents a modeling framework that incorporates those hypotheses. It also provides details on implementing various path generation algorithms, specifying the Path Size Logit and Logit Kernel models for route choice, and suggestions of explanatory variables for different components of the model framework. Chapter 4 describes the data collection and manipulation effort to prepare for model estimation. Chapter 5 summarizes model estimation results and recommendations for analysts wishing to apply the methodology presented to other settings. Chapter 6 provides a summary of the research contributions and a list of suggestions for further investigation.

2
Literature Review

Modeling travelers' awareness of routes draws from many disciplines. First, we consider existing transportation planning models. Understanding the assumptions that underlie these models and the limitations that result from these assumptions will help us identify other, less restrictive assumptions. We will also review the various techniques other authors have used to consider awareness of alternatives and the travel impact of information. This review will allow us to identify and expand on the most promising approaches. Since this route choice modeling problem involves the decisions of humans acting in spatial networks, we also examine relevant contributions from fields such as psychology, cognitive science and geography. Psychology, cognitive science and related fields provide insight into the human decision-making process. Researchers in these fields investigate questions such as how people perceive information and how people process that information to decide on a course of action. Geography is concerned with how space introduces complexities in problems that might otherwise be viewed from a purely economic or accounting perspective. Therefore, different configurations of the same quantity of activities and transportation facilities (that is, supposing some measure of supply, such as lane- or seat-miles is held constant) can result in different travel and activity patterns.

Accordingly, each discipline and topic is examined in a corresponding section. Existing transportation route choice models are described in section 2.1. Studies of the impact of information technologies on travel patterns using extensions of the four-step process or simulation are described in section 2.2. Section 2.3 provides a summary of the biological and behavioral theories regarding perception and decision-making. Section 2.4 considers the implications of current behavioral theories on the assumptions that underlie economic rational utility-maximization models, which form the basis of transportation choice models. Finally, some geographic theories related to transportation and results of geographical experiments are presented in section 2.5.

2.1 Existing Route Choice Models

This section classifies and describes some of the common techniques used by transportation practitioners for modeling route choice of auto users. While transit route choice is not explicitly considered here, many transit route choice (or “assignment”) techniques are extensions of the methods presented below. Readers interested in transit assignment may wish to consult Cascetta (2001); Lam (1999); Nuzzolo and Russo (1998); Jayakrishnan, McNally and Marar (1995); Nguyen and Pallottino (1994); Wu, Florian and Marcotte (1994); Spiess and Florian (1989); and Nguyen and Pallottino (1985). Most transportation modeling applications adopt a graph theoretic representation of the physical network. That is, intersections are represented as *nodes*, and roadways are represented as *links* or *arcs* (the terms are equivalent), which end at nodes. Table 2-1 shows sizes of some typical transportation networks for metropolitan areas. The networks described by Nuzzolo and Russo (1998) have nodes that represent locations in both space and time, rather than being purely spatial or static networks.

Links may be identified by their tail and head nodes, also called *A-Nodes* and *B-Nodes*, and may represent one- or two-way streets. Associated with each link is a collection of attribute values, such as distance, number of lanes, speed limit and congested travel time. Nodes could also have attributes associated with them, such as intersection delays or transfer waiting times on transit. These node attributes are easily implemented in a Geographical Information Systems (GIS) environment, such as TransCAD. However, earlier transportation planning software, such as Urban Transportation Planning System (UTPS, see USDOT, 1986) did not implement node attributes. Instead, UTPS associated these attributes with links or other data structures. These database structures have generally been maintained when network data is converted from UTPS to a GIS environment. Changing network representation typically involves new data collection initiatives by Metropolitan Planning Organizations (MPOs). Transportation models, such as those used for route choice, consist of operations on the links and nodes.

Metropolitan Area	Population	Zones	Nodes	Links
Boston, MA[1,2]	4,031,519	888	13,003	19,174
Cedar Rapids, IA[1,3]	179,411	537	2,600	3,800
Dallas-Ft. Worth, TX[4]	4,121,000	919	14,200	21,636
Montreal, QC[5,6]	3,428,300	699	6,207	19,304
New York, NY/NJ/CT[1,7]	19,013,777	2,396	16,236	24,628
Ottawa-Hull, ON/QC[5,6]	1,056,700	258	2,311	7,352
Portland-Vancouver, OR/WA[1,8]	1,758,937	1,244	9,728	25,166
St. Louis, MO/IL[1,9]	2,434,570	1,109	4,078	13,000
Tulsa, OK[10, 11]	574,241	524	5,380	10,000
Winnipeg, MB[5,6]	676,400	154	903	2,975

Notes: Populations of Canadian metropolitan areas are rounded to the nearest hundred.

The numbers of nodes and links for the Cedar Rapids network are rounded to the nearest hundred.

The population shown for Dallas-Fort Worth (rounded to the nearest thousand) is the North Central Texas Council of Governments' 1995 estimate for the 5,000-square-mile Metropolitan Planning Area represented in the transportation planning network. NCTCOG performs trip generation using 5,999 zones, which are aggregated to 919 zones for trip distribution, mode choice and traffic assignment. The number of nodes for the Dallas-Fort Worth network is rounded to the nearest hundred.

The numbers of links in the St. Louis and Tulsa networks are rounded to the nearest thousand.

Sources:
1. U.S. Bureau of the Census (1997) estimates for 1 July 1996.
2. Peterson (1998)
3. Horowitz and Granato (1999)
4. Cervenka (1999)
5. Statistics Canada (1999) estimates for 1 July 1998.
6. Chabini, Florian and La Saux (1996),
7. New York Metropolitan Transportation Council (1997)
8. Metro (1998)
9. East-West Gateway Coordinating Council (1997)
10. Indian Nation Council of Governments (1998) population estimates for 1999
11. Putta (1999)

Table 2-1. Sizes of Realistic Networks.

The following sections can be grouped into several categories describing the types of network algorithms and the behavior they represent. The first category involve how individual traveler choices interact with network characteristics (notably capacity) and result in an equilibrium between traveler volumes and travel times. Section 2.1.1 describes the shortest path problem, and section 2.1.2 the user equilibrium when travel times are known with certainty. Section 2.1.3 describes the equilibrium that results when travel times are stochastic, perhaps caused by traveler perception errors.

The next major category describes how alternative paths may be determined from network structure. Section 2.1.4 describes the "labeling" approach, which considers multiple traveler objectives, while section 2.1.5 discusses algorithms for identifying many attractive paths based on a single criterion. Specific heuristics to solve this problem are described in sections 2.1.6 through 2.1.8.

The next group (sections 2.1.9 through 2.1.12) focuses on several types of model formulations to produce more realistic predictions of path shares when some paths share common segments that might be perceived similarly.

The final two sections address how awareness may be modeled; section 2.1.13 presents an implicit approach where a term reflecting availability is added to each alternatives' utility, and section 2.1.14 describes an explicit choice set generation process. Section 2.1.15 compares the implicit and explicit approaches with a mathematical example.

Section 2.1.16 recapitulates the discussion of this section.

2.1.1 Shortest Paths / All-Or-Nothing

The simplest route choice models assume that travelers minimize a single variable such as distance or travel time. Such a setting is commonly called a "shortest path problem," and Dijkstra's algorithm (see Dijkstra, 1959, or the summary of algorithms in Ahuja, Magnanti and

Orlin, 1993) provides a well-established solution. Loading all the trips of an origin-destination pair on the links of the shortest path is commonly called an “All-Or-Nothing” traffic assignment.

Using a shortest path algorithm implicitly assumes that the traveler being modeled is aware of all the links (and their costs) that are used by the algorithm. Jan, Horowitz and Peng (2000) examined driver route choice data recorded by Global Positioning System (GPS) receivers, and present anecdotal evidence that the subject travelers in Lexington, Kentucky, did not select the shortest path. Those authors state that the data do not allow further analysis of why the drivers did not choose the shortest path.

2.1.2 User Equilibrium

When congested travel time is chosen as the objective variable, the problem becomes more complex, as congested travel time is a function of the volume of travelers using a link. The Bureau of Public Roads (BPR) formulation (see U.S. Department of Commerce, 1964) is commonly used in transportation planning applications. This formula computes congested times as

$$T_c = T_0 \left[1 + \alpha \left(\frac{V}{C} \right)^{\beta} \right]$$

where T_c is the congested travel time on a link,
T_0 is the free-flow travel time,
V is the hourly volume,
C is the hourly “practical” capacity, and
α and β are parameters. Typically, $\alpha = 0.15$ and $\beta = 4$, although in some instances other values – such as $\beta = 5.5$ for freeways – may be used.

Other formulations, such as those based on fundamental diagrams of traffic flow may also be used. Horowitz (1997) describes some of the challenges in using delay relations described by the 1994 Update to the *Highway Capacity Manual*. Horowitz also advocates using delay functions that take as arguments the volume on the link of interest and volumes on nearby links, to more accurately model the performance at signalized intersections, at two-way and four-way stops, and on rural two-lane roads where passing in the oncoming travel lane is permitted. Solving the user equilibrium problem when link travel times are functions of volumes on other links is discussed further below.

To solve the flow-dependent shortest paths problem, an iterative approach is often used, alternating between finding shortest paths for fixed travel times, and recalculating travel times based on new link volumes. An equilibrium results between travelers' choices of routes (which depend on travel time), and link traversal times (which depend on the volumes of drivers using them). When travelers, who seek to minimize travel time, have no incentive to change routes, the iteration calculations have reached a state corresponding to *Wardrop's* (1952) *first principle*. An alternative way of stating this principle is that for a given origin-destination pair, all used routes have the same travel time, and unused routes have travel times greater than or equal to that of the used routes. The situation described by Wardrop's first principle is often called *user equilibrium*, as it depends on individuals minimizing their own travel times. It is also common to see references to the *user optimal* rule in situations where equilibrium may not be guaranteed (for instance, in dynamic settings, or if the computation is halted before convergence is reached).

In comparison, *Wardrop's second principle*, which describes the situation where total travel time on the network is minimized, is often referred to as a *system optimal* assignment. To achieve a system optimal assignment, travelers must be assigned to links considering the *marginal cost* an additional traveler induces on link travel times, rather than the *average cost* described by the congested time of the BPR formula. Because the BPR formula (and other congestion relationships, such as those described by queuing models) is concave upwards, the marginal cost of travel will be greater than the average cost. The difference between marginal and average travel cost is a negative externality to other travelers, and thus, the network-wide travel time under user equilibrium will be greater than under a system optimal assignment. Some researchers have advocated that ATIS could be used to persuade travelers to use system-optimal routes. In contrast, Hall (1996) argues that democratic societies value honest travel information, and therefore travelers will resist system-optimal paths that are not personally optimal. Hall asserts that travelers who *perceive* information to be inaccurate will either ignore or act contrary to that information. Vaughn, Abdel-Aty, Kitamura, Jovanis and Yang (1993) and Bonsall and Joint (1991) present evidence of travelers responding contrary to recommendations given by inaccurate or misleading ATIS.

Any user equilibrium assignment routine can be used to produce a system optimal assignment by replacing average link costs with marginal costs. The marginal cost, *MC*, according to the BPR formulation can be shown to be

$$MC = T_0\left[1+\alpha(\beta+1)\left(\frac{V}{C}\right)^{\beta}\right].$$

Some transportation modeling packages allow users to supply their own values of the BPR parameters. Therefore, an analyst desiring a system-optimal assignment could replace the standard value of α, 0.15, with $\alpha(\beta-1)$ or 0.75.

Link delay relations can be described as “symmetric” or “asymmetric,” depending on the interrelationships of volumes on various links to corresponding travel times on those links. In the symmetric case, the impact of link A’s volume, V_A, on link B’s travel time, T_B, is identical (or symmetric) to the impact of link B’s volumes, V_B, on link A’s travel time, T_A. Mathematically, this may be written as

$$\frac{\partial T_B}{\partial V_A} = \frac{\partial T_A}{\partial V_B}.$$

The familiar case assumed by the BPR function, where a link’s travel time depends on only its volume, is a special case of symmetric delay relations. (In this case, the partial derivative terms $\frac{\partial T_B}{\partial V_A}$ and $\frac{\partial T_A}{\partial V_B}$ are zero for all pairs of links.) The “symmetric” designation therefore describes the property of the Jacobian matrix of travel time functions, that is, the square matrix of (partial) derivatives of the vector of link travel times with respect to the vector of link volumes. If a minimization formulation is desired, the objective function must be such that its second derivative (Hessian) matrix is the same as the Jacobian of travel time with respect to link volumes. Alternatively stated, the objective function may be any line integral of the link delay function. (For more details, see Section 8.1 in Sheffi, 1985.)

The asymmetric case is therefore one in which one link's volumes may have a greater affect on another link's travel time than that link's volume has on the first link's time. An example of a situation in which a non-symmetric volume-delay function would be appropriate is a ramp merging onto a busy freeway. Cars on the ramp must wait for a gap in the mainline freeway traffic in order to merge, and therefore leave the ramp. For a link representing the ramp, we would model its traversal time as a function of both the ramp volume and the volume of the mainline link upstream of the merge point. However, because ramp traffic must yield to mainline traffic, the traversal time of the upstream mainline link depends only on its volume, and not on the ramp volume. A similar argument may be presented for a cross street approaching a major arterial at a two-way stop.

Traffic assignment for networks with asymmetric volume-delay functions is often solved by a technique called *diagonalization*. Since an equivalent minimization formulation doesn't exist for the asymmetric case, an approximate objective function is used, by assuming that flows on other links are fixed when calculating a link's travel time. This approximation of course results in a longer, but still feasible, computational time.

Two approaches for solving the flow-dependent traffic assignment problem are the method of convex combinations – popularly called the Frank-Wolfe Algorithm – and the Method of Successive Averages (MSA).

The method of convex combinations was first proposed by Frank and Wolfe (1956) as a general procedure for solving a nonlinear optimization problem by decomposition. The problem is transformed into a linear program and a one-dimensional non-linear problem or "line search."

The linear program step is called “direction finding,” as it represents a search for a new feasible solution by which to improve the objective function. The line search step determines weights to average the current direction-finding solution with previous results to obtain a new minimum of the objective function. Bruynoughe (1968) was the first to propose applying this method to the traffic assignment problems. In this problem, the direction-finding step is solved by a shortest path calculation (using for example, Dijkstra’s algorithm) assuming fixed travel times.

The MSA (see Almond, 1967) is a more general and robust approach than the Frank-Wolfe algorithm. It maintains the direction-finding step, but instead of calculating a weight in a line-search step, the MSA uses predetermined, fixed weights. Therefore, the MSA, unlike the Frank-Wolfe Algorithm, does not requires a minimization formulation, and may therefore be easier to program. Sheffi (1985) discusses the regularity conditions under which the MSA is guaranteed to converge to a solution. However, for problems where the Frank-Wolfe Algorithm can be used, the MSA is often slower, because it cannot take advantage of optimized step sizes from the line search routine. Recall that for the traffic assignment problem, a minimization formulation exists for symmetric link travel time cases, and in asymmetric cases, the diagonalized problem is often solved. For a more detailed discussion of the use of the MSA and Frank-Wolfe algorithms in more complicated transportation settings (e.g., dynamic traffic assignment), see Bottom (2000).

2.1.3 Stochastic User Equilibrium / Multinomial Logit

Stochastic assignment was developed to relax the assumptions of all-or-nothing shortest path assignment used for (deterministic) user equilibrium models. Stochastic assignment often adopts the multinomial logit (MNL) model for spreading travelers among different feasible path. MNL models assume that travelers have the same error distribution in the utility term, based on the Type I Extreme Value distribution. (This probability distribution is also called the Gumbel distribution, and is occasionally confused with the Weibull distribution.) In the case of stochastic assignment, these errors are theorized to result from perception errors of travelers. This model is commonly written

$$P(i) = \frac{e^{-\theta L_i}}{\sum_{j \in C_n} e^{-\theta L_j}}$$

where $P(i)$ = the probability of a traveler using path i, which can also be interpreted (assuming homogenous traveler characteristics) as the proportion of trips in an origin-destination pair using path i,

C_n = the choice set of feasible paths for individual n,

L_i, L_j = the length (impedance) of paths i and j, respectively, and

θ = a utility coefficient or "spread parameter."

To use conventional MNL software, it is necessary to identify or "enumerate" the possible paths, and to provide values of attributes (such as distance, travel time, tolls) of those paths. In realistic networks, there may be an inordinately large number of possible paths, although travelers may consider a much smaller number of attractive paths. Sections 2.1.4 through 2.1.8 discuss some explicit enumeration methods.

Dial (1971) developed the STOCH algorithm to assign trips among links according to the MNL formulation without having to explicitly identify or "enumerate" the possible paths in C_n. The STOCH algorithm uses a choice set consisting of all paths using only links that would take a traveler further away from the origin and closer to the destination. Such paths may be called "efficient" or "reasonable" in the stochastic traffic assignment literature.

As in the case of deterministic user equilibrium, an iterative procedure can be used to achieve consistency between the travel times assumed during stochastic loading and the travel times that would result from applying a congestion relation to those assigned flows. The resulting consistency is called stochastic user equilibrium (SUE). Fisk (1980) presented a minimization formulation for solving the SUE problem, and Chen and Alfa (1991) developed a convex combinations (Frank-Wolfe) algorithm to gain faster convergence of the SUE algorithm than that offered by MSA approaches. In each iteration, the STOCH algorithm is used to assign travelers to links, then congested times are recalculated and averaged with those of previous iterations.

2.1.4 The Labeling Approach

One technique for generating multiple possible paths that might be considered by those traveling from a given origin to a given destination is to consider that different travelers may have different objective functions in seeking routes. Some drivers may wish only to minimize travel time. Others may feel uncomfortable making difficult maneuvers, and therefore avoid lane changes, freeways, heavily-congested roads or left turns at intersections without protected signals. Still others, perhaps making trips for non-work purposes or with considerable arrival time flexibility, may seek out scenic routes. Each of these criteria may correspond to a different route being preferred, and thus, each route can be "labeled" by the criterion (or criteria) for which it is optimum. This approach was proposed by Ben-Akiva, Bergman, Daly and Ramaswamy (1984), and the objective functions or labels they used are summarized in Table 2-2.

Label	Link Impedance
Minimize Time	Time
Minimize Distance	Distance
Maximize Scenery	Time (1 + β_1 percent non-scenic)
Minimize Traffic Lights	Time + β_2 number of traffic lights
Minimize Congested Travel	Time (1 + β_3 high V/C dummy)
Maximize Use of Highways	Time (1 + β_4 non-highway dummy)
Maximize Use of High-Capacity Roadways	Time (1 + β_5 low capacity dummy)
Maximize Travel Through Commercial Areas	Time (1 + β_6 non-commercial dummy)
Maximize Road Quality	Time (1 + β_7 low quality dummy)
Hierarchical Travel Pattern (Three Levels Considered)	Time (1 + β_{81} level 1 dummy + β_{82} level 2 dummy)

Source: Ben-Akiva, Bergman, Daly and Ramaswamy (1984) Table 2.

Table 2-2. Examples of Route Labels and Corresponding Objectives.

It is possible for two or more labels to refer to the same physical path. The nested logit model is used in this situation, so it will be helpful to first review the nested logit (NL) model. A more thorough discussion of the nested logit model is available in Ben-Akiva and Lerman (1985).

Nested logit models are most frequently seen in mode choice and in multi-dimensional choice, such as combined destination and mode choice. In mode choice, nested logit models are used when alternatives can be placed into groups with common unobserved attributes or groups in which members are closer substitutes for each other than members of other groups. For example, in the urban mode choice context, auto driver, auto passenger and taxi passenger may be placed in the “private transportation” group, while bus, subway and commuter rail belong to the “public transportation” group. In multi-dimensional choice, the outcome of one choice determines the grouping. These groups are the “nests” that the nested logit name refers to. Nested logit is also sometimes called tree logit, after the hierarchical figures that are commonly used to illustrate the grouping of alternatives. Examples of these figures are shown in Figure 2-1 for mode choice and Figure 2-2 for combined residence and mode choice. Figure 2-2a shows a schematic for a model where mode choice is conditional on residence choice. Such a structure would be appropriate for a household with access to a car and in which members consider whether auto or transit would be preferred for each trip made. Figure 2-2b shows residence choice conditional on mode choice. Such a model structure would be appropriate for a household making the lifestyle decision between owning an auto or committing to use transit for all trips, and then selecting a residence that would best allow members to make trips using the preferred or habitual mode. For example, a household deciding to own an auto may prefer a house in suburban locations where parking is more plentiful and traffic congestion may not be as severe as in the center city. In contrast, a household wishing to use transit for the bulk of its trips might prefer to reside in close proximity to the rapid transit system. Mathematically, we can describe partitioning the choice set C_n into M nests C_{mn} such that

$$C_n = \bigcup_{m=1}^{M} C_{mn}$$

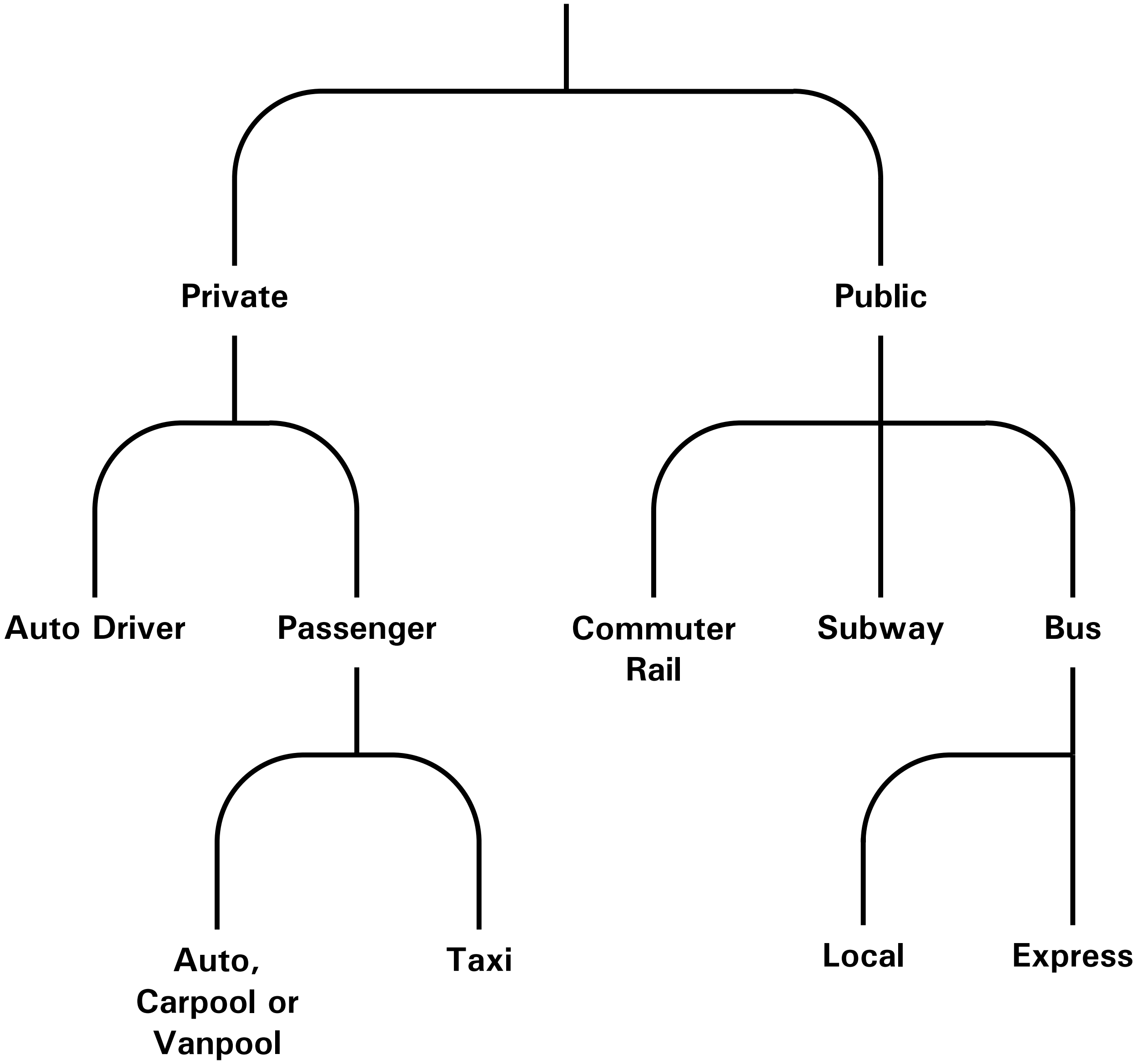
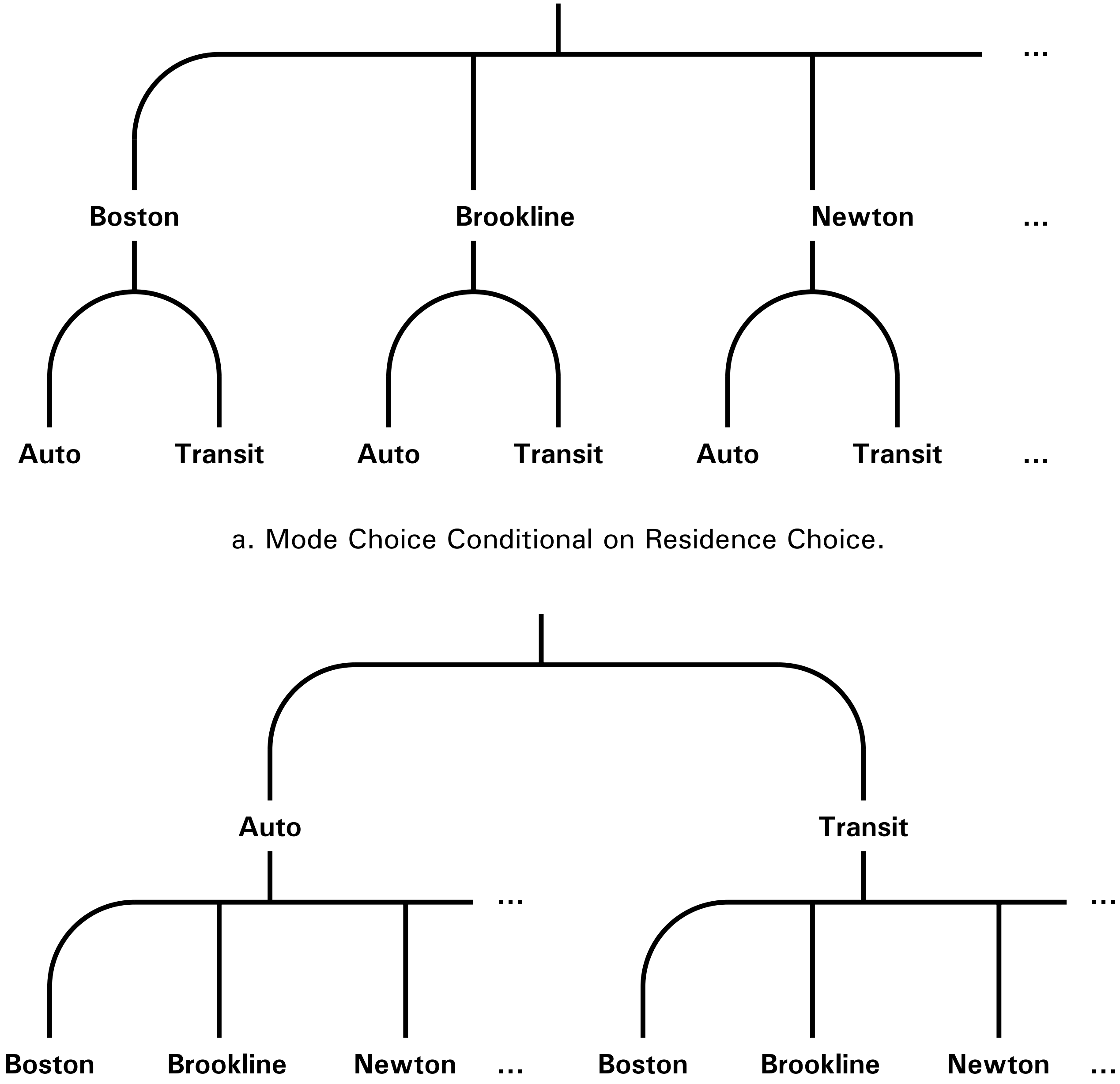


Figure 2-1. Schematic of a Nested Mode Choice Model.



a. Mode Choice Conditional on Residence Choice.

b. Residence Choice Conditional on Mode Choice.

Figure 2-2. Two Hierarchies for a Nested Multidimensional (Mode and Residence) Choice Model.

and

$C_{mn} \cap C_{m'n} = \varnothing \;\; \forall\, m \neq m'$.

The probability of choosing an alternative is the product of choosing its nest, and choosing that alternative from within the nest; that is,

$$P(i|C_n) = P(C_{mn}|C_n)\,P(i|C_{mn}) \tag{2-1}$$

Within the nest, the choice probability of an alternative is based on multinomial logit:

$$P(i \mid C_{mn}) = \frac{e^{V_{in}}}{\sum_{j \in C_{mn}} e^{V_{jn}}}, \tag{2-2}$$

where V_{in} is the systematic utility of alternative *i* to traveler *n*.

The utility of a nest is based on the combination of the utilities of the alternatives within the nest, $I_{C_{mn}}$, and utility from attributes of the nest that cannot be assigned to a single lower-level alternative, $\widetilde{V}_{C_{mn}}$.

$$V_{C_{mn}} = \widetilde{V}_{C_{mn}} + \mu I_{C_{mn}} \tag{2-3}$$

For example, in a model with the structure shown in Figure 2-2a, the nest utility will be a function of the attributes of the residence zone. The combined utility of the alternatives in the nest is based on the *expected value of the maximum utility* of those alternatives. Ben-Akiva and Lerman (1985) have shown that the expected maximum utility can be calculated by the "log-sum" formulation:

$$I_{C_{mn}} = \ln \sum_{j \in C_{mn}} e^{V_{jn}} \tag{2-4}$$

This term is also called *inclusive value* or *inclusive price*, since it reflects the utility of all of the alternatives in a nest. The inclusive value of a destination choice nest is often used as an accessibility measure. (See Walker, 2000, or Ramming, 1994, for examples.) Discrete choice theory suggests that the coefficient on inclusive value (sometimes called the “nesting coefficient”) should be between 0 and 1. A value of 0 suggests two independent or unrelated decisions. A value of 1 means the nested logit model reduces to multinomial logit. A nesting coefficient value greater than 1 generally suggests that another nesting structure is appropriate, for instance, reversing the hierarchical order of a multi-dimensional choice.

In the labeling method, labeled paths are organized into a nested logit structure, with physical paths forming the upper-level nest. That is, labeled paths are grouped according to physical paths. However, travelers do not choose among labels, but rather among physical paths. This is not the typical nested logit estimation setting, where the choice of lowest-level alternative is known. (For example, in the mode choice situation of Figure 2-2, we may know that a particular traveler chose taxi passenger in the private transportation nest.) Estimating the labeling model involves analytically calculating the formulation of the inclusive values and constructing a custom maximum likelihood estimation program to determine the coefficients of the non-linear-in-parameters model suggested by equations 2-1 through 2-4. Ben-Akiva, Bergman, Daly and Ramaswamy (1984) provide several specifications and discuss identification of model parameters. The authors consider path attributes and label-combination dummies (e.g., 1 if the physical path corresponds to exactly four labels including the hierarchical route label) in the physical path nest, and only label-specific constants at the labeled path level. If the labeled paths are assumed to have the same utility (excluding physical path terms), this technique is similar to the size correction used for aggregate alternatives, including the Path Size Logit model described in section 2.1.10 below.

2.1.5 K-Different Path Algorithms

K-Different Path algorithms are generalizations or repeated applications of shortest path algorithms that generate a collection of paths. Since the additional paths have impedances close to the impedance of the shortest path, one would expect these to be among the routes travelers might consider. One class of these algorithms use an exact solution to the problem of finding the *K* routes with the minimum values of a given objective function (e.g., minimize distance, minimize free-flow time, minimize congested time). Heuristic solutions are also common, and may be classified into three groups: (1) Elimination techniques remove links from shortest paths on the network in order to identify more short paths. (2) Overlapping penalty approaches increase the impedance of links on the shortest path, but do not remove them from the network, before searching for more short paths. (3) Branching methods select a link branching off a previously-identified short path, and construct a new path by finding the shortest path from the origin to the links' tail node, and from the links' head node to the destination.

Several exact solutions to the K-Shortest Path problem have been developed. (See for example Ziliaskopoulos, 1994; Shier, 1979; Dreyfus, 1969; Bellman and Kalaba, 1968; Pollack, 1961; and Hoffman and Pavley, 1959.) These algorithms are generally extensions of the label-setting and label-correcting approaches used to determine a single shortest path, such as Dijkstra's Algorithm. However, instead of maintaining a single label at each node, an array of *K* labels are used, so that paths may be sorted in ascending order of length, travel time or other objective.

Authors such as Park and Rilett (1997) and Scott, Pabón-Jiménez and Bernstein (1997) have questioned the usefulness of these algorithms in generating possible routes to provide through ATIS, because the paths these algorithms generate tend to be very similar, while paths that a human driver would consider (perhaps in other corridors) are not identified. As an example, consider an origin-destination pair where the shortest path passes through a rotary. (Rotaries are called traffic circles or roundabouts outside New England.) An exact K-Shortest Path algorithm that allows looping might select the shortest path plus a full revolution in the rotary as the second-best path. The third-best path would involve two revolutions in the rotary, and so on. Even when K-Shortest Path algorithms do not allow looping, next-shortest paths may be built that represent minor deviations from earlier paths, especially in areas of considerable network roadway density.

We examine heuristics to approximate the K-Shortest Path solution in the following three sections. These heuristics may be classified according to how new alternative paths are identified. The technique described in section 2.1.6 starts with the shortest path and removing – or "eliminating" – some links on that path from the network. The "link penalty" approach described in section 2.1.7 increases the impedance of links on the shortest path, but does not prevent them from being used in successive paths. Finally, random draws of link attributes may be made to reflect travelers' perception errors or an unreliable travel environment. (See section 2.1.8.)

2.1.6 Link Elimination Approaches

The elimination approach operates by iteratively identifying the shortest path, removing all – or some of – the links on that path from the network, and finding the new shortest path. Azevedo, Costa, Madeira and Martins (1993) describe one algorithm where all the links used by the shortest path are removed from the network to find the next-best path. One danger of eliminating all (or even many) of the links on the shortest path at once is that removing centroid connectors (artificial links that connect a representative node for each origin – the *centroid* – to the "real" network links) and major crossings may make it infeasible to construct more paths between the origin and destination. While network disconnection is used as a completion criterion, it may be reasonable for travelers to have alternative paths to a particular major crossing. These paths using alternative access to a major crossing would not get generated under this type of procedure.

Another variant of this approach eliminates one link at a time, but this may result in the minor deviation problem described above. It might be possible to generate alternative paths by eliminating individual links from the shortest path, then eliminating all combinations of pairs of links, and so on. However, realistic paths may contain up to 300 links. With this number of links, the possible combinations of links to be eliminated becomes quite daunting. Further, individual or combinations of links may also be eliminated from the second, third and following paths generated by this method.

A heuristic related to the elimination approach may be called “branching,” and was first proposed by Bellman and Kalaba (1968). It uses the following procedure: First, the shortest path in the network is identified. Then a link with one node on the shortest path and its other node off the path is selected from among all such links. Such links may be examined in order from origin to destination, or by taking the link with the highest capacity, etc., first. A new path constrained to use this link is then constructed. This new path may be generated by taking the shortest path from the origin to the tail of the selected link, and the shortest path from the head of the link to the destination. This path is then added to a data structure where paths are sorted in order of increasing length. Additional paths may be generated by using other links to branch off the shortest path. Paths may also be generated by branching from the second-shortest path, and so on.

2.1.7 Link Penalty Approaches

Instead of eliminating links from consideration, overlap penalty approaches increase the impedance on links used by the previously-identified shortest paths when searching for new paths. This has the advantage of allowing essential links to still be used in later paths, while discouraging the use of already identified links where alternatives exist. That is, while the link elimination approach could result in a disconnected network, the overlap penalty approach would still allow a (very high impedance) path to be built. De la Barra, Perez and Anez (1993) describe a technique by which the shortest path is identified, impedance on those links are increased by a fixed percentage, and the shortest path calculation repeats. Park and Rilett (1997) modify the de la Barra approach by not increasing the impedance on links within a certain distance (which may be measured on the network or as an airline distance) from the origin or destination, to avoid producing minor deviations at the start or end of the route. Travel time may also be used to determine which links near the origin and destination should be exempt from having their impedance increased. Scott, Pabón-Jiménez and Bernstein (1997) present an optimization program for determining how much to increase the impedance on shortest path links in order to generate a next-shortest path that overlaps with the shortest path by no more than a given number of links.

2.1.8 Simulation Approaches

The K-Shortest Path algorithms mentioned in the previous sections all make use of a single link attribute. It is possible to perform multiple runs of these algorithms using different attributes as the basis – for example, first using distance, then free-flow time, and finally estimated time as the objective function for the first shortest path. However, these attributes are often correlated. Further, paths generated by these K-Different Path methods are often very similar, and may not capture the full range of paths that human drivers may consider. Because analysts posit that drivers perceive travel times with error, a reasonable approach would be to make random draws from a distribution that might represent drivers' perceptions.

None of the K-Shortest Path references cited in the previous sections discuss simulation as a method of generating alternative routes. Sheffi and Powell (1982) describe a Monte Carlo technique used to apply the Multinomial Probit model to traffic assignment, because the Gaussian distribution does not have a convenient analytical solution for this problem. (See Section 2.1.12 for further discussion of the Multinomial Probit model.) Sheffi and Powell's technique involves a preset number of iterations, during which realizations of link travel times are drawn from the Probit distribution. All-Or-Nothing assignment is used to load trip volumes onto network links. At the end of the algorithm, flows from all iterations are averaged to produce the final forecast of link flows. For path generation applications, it is desirable to save the links on the shortest path at each iteration rather than the averaged link flows.

A comparison between User Equilibrium assignment and the link penalty heuristic may be useful here. Consider a User Equilibrium assignment procedure involving many iterations of All-Or-Nothing assignment to shortest paths based on costs calculated from link flows predicted by the previous iteration. If we neglect the step of loading link flows, we can imagine that the link penalty approach – in which links on the shortest paths have their impedance increased by a fixed amount instead of according to the BPR equation – is a crude version of the User Equilibrium algorithm. An analogy may be made to the simulation technique being the Sheffi and Powell assignment algorithm without the network loading steps.

Sheffi and Powell's technique was developed for applying a Probit model where the distributional and utility parameters were already estimated. Since we desire to generate paths from which to estimate a route choice model, the distributional parameters from which simulations are drawn must come from another source. These parameters may be chosen from a model of travel time perceptions, or "calibrated" by choosing values to maximize the "coverage" or otherwise produce paths with desired properties. The process of choosing parameters for the simulation method will be discussed further in Chapter 3.

2.1.9 C-Logit

One limitation of logit formulations is the assumption that error terms are independent and identically distributed (i.i.d.) Gumbel, which results in the Independence from Irrelevant Alternatives (IIA) property. For route choice, this property is often illustrated by the three routes presented in Figure 2-3 below. The three routes all have the same distance (or impedance), T. Paths 1 and 2 share a common segment, with a length (or impedance) of $T - d$, and are distinct for d units. Assuming that route utility is based on distance only, and since the three routes all have the same distance, multinomial logit (MNL) models will predict a share of one-third for each of the routes. MNL is consistent with our intuition when the overlap between Paths 1 and 2 is infinitesimally small (that is, as $d \to T$). However, when the overlap approaches the length of the whole route ($d \to 0$), we expect that Path 3 would have a share of one-half, while Paths 1 and 2 (which become difficult to distinguish) would each receive one-quarter of the traffic. For intermediate degrees of overlap, we expect Paths 1 and 2 to have shares between one-quarter and one-third.

Cascetta, Nuzzolo, Russo and Vitetta (1997) proposed the C-Logit model to maintain the computational simplicity of the logit form, but produce more intuitive forecasts of route shares.

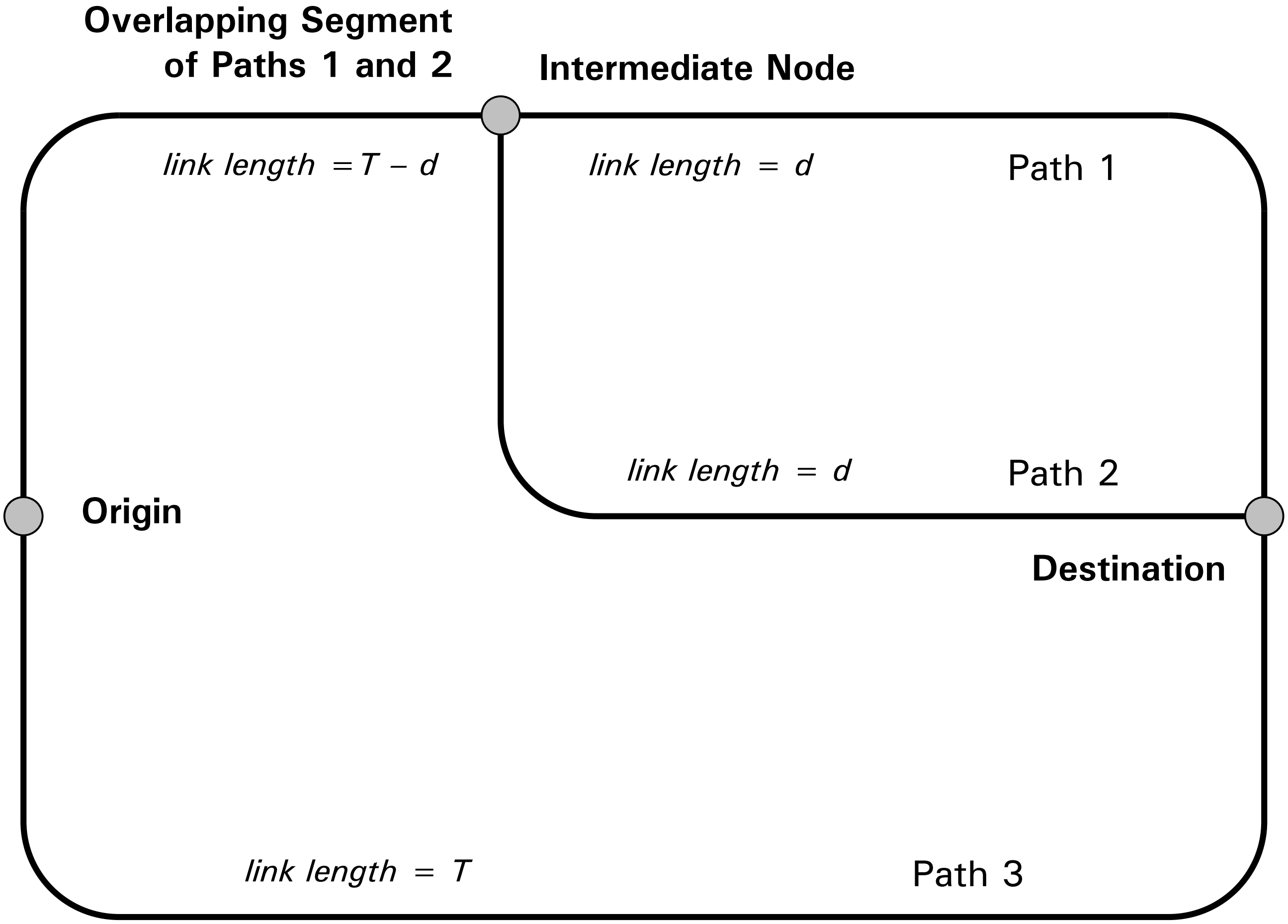


Figure 2-3. The Overlapping Path Problem.

The C-Logit model adds an adjustment to route utilities based on the amount of overlap with other routes. This correction term is called a "commonality factor," *CF*, and enters the logit form as

$$P(i \mid C_n) = \frac{e^{V_{in}+CF_{in}}}{\sum_{j \in C_n} e^{V_{jn}+CF_{jn}}}.$$

Note that the value of *CF* should always be negative, because overlapping paths will receive lower shares than if they were unique (that is, the shares predicted by MNL).Often, the commonality factor term takes a logarithmic form, so that in the case of a unique path, the log transform has an argument of one, and therefore, no adjustment to path utility occurs. For paths with overlap, the argument is less than one and possibly approaching zero, so that such paths utilities appear less attractive.

The authors propose four different forms for the commonality factor correction:

$$CF_{in} = -\beta_0 \ln \sum_{j \in C_n} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right)^{\gamma}, \tag{2-5}$$

$$CF_{in} = -\beta_0 \ln \sum_{a \in \Gamma_i} \frac{l_a}{L_i} N_{an}, \tag{2-6}$$

$$CF_{in} = -\beta_0 \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \ln N_{an}, \text{ and} \tag{2-7}$$

$$CF_{in} = -\beta_0 \ln \left[1 + \sum_{\substack{j \in C_n \\ j \neq i}} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right) \left(\frac{L_i - L_{ij}}{L_j - L_{ij}} \right) \right], \tag{2-8}$$

where β_0 and $\gamma$are coefficients to be estimated or calibrated,

L_{ij} is the length paths i and j have in common,

Γ_i is the set of arcs in path i,

l_a = the length of link a,

$N_{an} = \sum_{j \in C_n} \delta_{aj}$ for real links, 1 for centroid connectors, and

δ_{aj} = the link-path incidence dummy, that is, 1 if path j uses link a and 0 otherwise.

By considering some theoretical networks and comparing the predictions of the C-Logit model against Probit (see section 2.1.12), the authors conclude that the specification in equation 2-5 most resembles the Probit shares, although the value of γmust be calibrated. When path utilities are similar, the authors note that the specification in equation 2-6 gives similar results to the specification in equation 2-5. The lack of theory or guidance to which form of commonality factor should be used is a drawback of the C-Logit method.

Cascetta, Nuzzolo, Russo and Vitetta then estimate the C-Logit model for the choice of inter-city route by truck drivers in Italy, using the commonality factor given by equation 2-6.

2.1.10 Path-Size Logit

The Path-Size Logit model is an application of discrete choice theory for aggregate alternatives, which has been used in other transportation settings such as destination choice. Alternatively, Path-Size Logit (or PS Logit) represents an effort to incorporate behavioral theory in the C-Logit adjustment process. As in the C-Logit Model, Path-Size Logit adds a correction term to the utility of alternative routes:

$$\mathrm{P}(i \mid C_n) = \frac{e^{V_{in} + \ln PS_{in}}}{\sum_{j \in C_n} e^{V_{jn} + \ln PS_{jn}}} = \frac{PS_{in} e^{V_{in}}}{\sum_{j \in C_n} PS_{jn} e^{V_{jn}}}$$

where PS_{in} is the size of path i for person n.

A path with no overlapping links needs no utility adjustment and has a size of one. The limiting case of two paths being created by “duplicating” or “splitting an existing path down the middle” have a size of one-half each. This is the case in Figure 2-3 when d approaches 0. More generally, when one unique path is split into J duplicate paths, each resulting path has a size of $1/J$. The size of partially-overlapping paths can be thought of as composed of sizes of links, which are then weighted by some appropriate measure, such as a link’s percentage contribution to total path length (impedance). However, care must be taken when links are “split” among paths of differing lengths.

Path-Size Logit was introduced by Ben-Akiva and Ramming (1998), who presented the following formulation:

$$PS_{in} = \sum_{a \in \Gamma_i} \left(\frac{l_a}{L_i} \right) \frac{1}{N_{an}} = \sum_{a \in \Gamma_i} \left(\frac{l_a}{L_i} \right) \frac{1}{\sum_{j \in C_n} \delta_{aj}}$$

Note that the term (l_a / L_i) is a weight corresponding to the fraction of path impedance coming from a specific link. The remaining term, 1 / N_{an}, is based on the count of paths using the link. This term is one for links that are used by only one path; we refer to these links as "distinct" or "unique" segments of the path.

For links used by more than one path, this second term is not affected by the length or impedance of the paths using it. This formulation can therefore suffer when arbitrarily long paths are included in the choice set. For example, consider the case where the upper-right link of Figure 2-3 becomes very large. The second term for the upper-left link remains ½ for both Paths 1 and 2. Path 2 has a size of less than one. Since most of Path 1 is the distinct upper-right link, its size will be very close to one. Path 3 is distinct, and therefore has a size of 1. When we consider the choice probabilities, Path 1 will have a very small share because of its great length. Path 2 and 3 have the same length, and Path 2 has a size smaller than Path 3. Therefore, this model would predict that Path 2 has a smaller share than Path 3 even though Path 1 is essentially unchosen. This result will be further examined in Section 3.1.3.

2.1.11 Cross-Nested Logit

The Link-Nested Logit model was first proposed by Vovsha and Bekhor (1998) as an application of the Cross-Nested Logit (CNL) model to route choice. The Cross-Nested Logit model, Multinomial Logit model (including its variants, C-Logit and PS Logit) and Nested Logit model (including the application of nested logit to the labeling approach) are members of the broad Generalized Extreme Value (GEV) model. Assumptions and properties of the GEV model are discussed in Ben-Akiva and Lerman (1985), and are beyond the scope of this discussion. The

Cross-Nested Logit model differs from the Nested Logit model in that lower-level alternatives may belong to more than one nest. That is, we define a set of parameters for each alternative *i* and each nest *m*, parameters α_{mi} $(0 \leq \alpha_{mi} \leq 1)$, which represents the degree of “membership” or the inclusion weight of alternative *i* in nest *m*. The sum of α_{mi} over all nests is generally normalized to one for each lower-level alternative, *i*. The choice probabilities of the Cross-Nested Logit model are as follows:

$$P(i \mid C_n) = \sum_{m=1}^{M} P(C_{mn} \mid C_n) P_n(i \mid C_{mn}),$$

$$P(i \mid C_{mn}) = \frac{\alpha_{mi} e^{V_{in}}}{\sum_{j \in C_{mn}} \alpha_{mj} e^{V_{jn}}}, \text{ and}$$

$$P(C_{mn} \mid C_n) = \frac{e^{V_{C_{mn}} + \mu_m I_{C_{mn}}}}{\sum_{l=1}^{M} e^{V_{C_{ln}} + \mu_m I_{C_{ln}}}}.$$

For the Cross-Nested Logit model,

$$I_{C_{mn}} = \ln \sum_{j \in C_{mn}} \left(\alpha_{mj} e^{V_{jn}} \right)^{1/\mu_m}$$

Combining terms gives

$$P_n(i \mid C_n) = \frac{\sum_{m=1}^{M} \left(\alpha_{mi} e^{V_i} \right)^{\frac{1}{\mu_m}} \left(\sum_{j \in C_{mn}} \left(\alpha_{mj} e^{V_i} \right)^{\frac{1}{\mu_m}} \right)^{\mu_m - 1}}{\sum_{m=1}^{M} \left(\sum_{j \in C_{mn}} \left(\alpha_{mj} e^{V_i} \right)^{\frac{1}{\mu_m}} \right)^{\mu_m}}.$$

The Cross-Nested Logit model can be shown to reduce to MNL when all of the parameters μ_m are one:

$$P_n(i \mid C_n) = \frac{\sum_{m=1}^{M} \left(\alpha_{mi} e^{V_i}\right) \left(\sum_{j \in C_{mn}} \left(\alpha_{mj} e^{V_i}\right) \right)^{\mu_m - 1}}{\sum_{m=1}^{M} \left(\sum_{j \in C_{mn}} \left(\alpha_{mj} e^{V_i}\right) \right)} = \frac{\sum_{m=1}^{M} \left(\alpha_{mi} e^{V_i}\right)}{\sum_{j \in C_{mn}} \sum_{m=1}^{M} \left(\alpha_{mj} e^{V_i}\right)} = \frac{e^{V_i} \left(\sum_{m=1}^{M} \alpha_{mi} \right)}{\sum_{j \in C_{mn}} e^{V_i} \sum_{m=1}^{M} \left(\alpha_{mj}\right)},$$

and given that the alpha terms sum to one (or to the same constant for all alternatives *j*), this reduces to the MNL probability model.

In the Link-Nested Logit model, links (indexed by *a*) form the nesting scheme, while routes (indexed by *i*) form the lower-level alternatives. Vovsha and Bekhor calculate the inclusion weights as

$$\alpha_{ai} = \frac{l_a}{L_i} \delta_{ai}.$$

where l_a = the length (or time) of link *a*,

L_i = the length (or time) of route *i*, and

δ_{ai} = the link-route incidence dummy, that is, $\delta_{ai} = 1$ when route *i* traverses link *a*, and 0 otherwise.

The largest network considered by Vovsha and Bekhor (1998) contains one origin-destination pair, eight nodes (including the two centroids), eleven links, and five routes. For a realistic size network, and a realistic number of links per path, the nesting structure could become quite complex and therefore computationally onerous. Papola (2000) estimated a CNL model for inter-city route choice with a limited number of alternative routes. Bierlaire (2001) estimated a CNL mode choice model using inter-city data. Bierlaire’s model considered three modes, and used techniques to combine Revealed Preference (that is, observed) and Stated Preference choice indicators. Bierlaire states that the CNL model has better fit than a Nested Logit model, but adds that the assumption regarding normalization of α's has not been verified.

Note that Vovsha and Bekhor (1998) and Papola (2000) estimated CNL models using constant values of μ. Prashker and Bekhor (1998), Papola (2000), and Wen and Koppelman (2001) cite difficulties of making this assumption. Nest-specific μ's may be estimated if there is sufficient data for their identification. Bekhor (2001) proposed the following formulation based on path topology:

$$\mu_a = 1 - \frac{\sum_{j \in C_n} \alpha_{ai}}{\sum_{j \in C_n} \delta_{ai}}. \qquad (2\text{-}9)$$

Swait (2001) proposes the Choice Set Generation Logit (GenL) model, in which choice sets form the nests of a CNL structure. Swait presents estimation results from inter-city data involving the choice among four modes. The author also acknowledges the computational difficulties of estimating a GenL model when the choice set is large.

We are unaware of any application of the Cross-Nested Logit model to even a moderate size city. (Note that Wen and Koppelman refer to the model with nest-specific μ's as "Generalized Nested Logit" while we retain the name "Cross-Nested Logit" for the general model.)

Vovsha and Bekhor compared the outcome of the CNL model against two common traffic assignment techniques – deterministic user equilibrium and (logit-based) stochastic user equilibrium. They established that for uncongested networks, the CNL results were quite different, but these differences became less pronounced in congested networks.

2.1.12 Probit and Logit Kernel

Since the IIA property of logit makes it difficult to represent the effect of overlapping paths, some researchers have examined the suitability of the probit model for route choice. Because the probit model is based on error terms having a multivariate normal distribution – as opposed to a Type I Extreme Value distribution as assumed in MNL and other GEV models – an arbitrary covariance structure may be specified. Daganzo (1977) was one of the first to use the multinomial probit model.

Yai, Iwakura and Morichi (1997) provide a recent example of an application of the probit route choice model in Japan. The authors assume the covariance of route utilities is proportional to overlap length. Routes are also assumed to have heteroskedastic error terms where variance is proportional to route length or impedance.

The difficulty in implementing the probit model is that no closed form exists for the Gaussian cumulative distribution function (CDF), so numerical techniques must be used. Numerical integration techniques are computationally feasible when the number of Gaussian variables (generally the number of alternatives less one, which is normalized to be the base alternative) is small. Hajivassiliou, McFadden and Ruud (1996) present some alternative estimation methods, and Bolduc (1999) advises that maximum simulated likelihood estimation with a Geweke-Hajivassiliou-Keane (GHK) probability simulator is the preferred method for transportation modeling with large samples and choice sets.

Choice models with combinations of Gaussian and Type I Extreme Value error terms have been proposed by researchers such as McFadden and Train (1998), who call the resulting model Mixed Logit, and by Ben-Akiva and Bolduc (1996), who refer to the resulting system as Multinomial Probit with Logit Kernel, or simply Logit Kernel. Other authors may refer to this formulation as Hybrid Logit.

The general form of the Logit Kernel model (in vector notation) is given by Walker (2000) as:

$$\mathbf{U} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} = \mathbf{X}\boldsymbol{\beta} + \mathbf{F}\mathbf{T}\boldsymbol{\zeta} + \boldsymbol{\nu}$$

where $\mathbf{U}$ is a J_n by 1 vector of utilities;

$\boldsymbol{\beta}$ is a column vector of K unknown parameters;

$\mathbf{X}$ is a J_n by K matrix of explanatory variables;

$\boldsymbol{\zeta}$ is a column vector of M i.i.d. standard Normal variables, which represent unobservable factors;

$\mathbf{F}$ is a J_n by M factor loading matrix (to be determined);

$\mathbf{T}$ is an M by M lower triangular matrix of unknown parameters (to be determined); and

$\boldsymbol{\nu}$ is a J_n by 1 vector of i.i.d. Gumbel variables with scale parameter μ.

Therefore,

$$\mathbf{Var}(\boldsymbol{\varepsilon}) = \mathbf{FTT}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}} + (g/\mu^2)\mathbf{I}_{J_n}$$

where g is the variance of a standard Gumbel variable, that is, π^2 / 6.

Elements of **F** and **T** may be estimated or specified from data. The advantage of logit kernel over pure probit is that **F** and **T** may be specified so that if cross-alternative correlations are estimated to be zero, the model reduces to MNL. Different specifications of **F** and **T** may be used for different purposes.

For example, to introduce a random coefficient on the k^{th} explanatory variable, one would use $\mathbf{F} = \mathbf{X}_k$ and $\mathbf{T} = \gamma_k$, where γ_k is a parameter related to the spread of β_k. Han, Algers and Engelson (2001) estimate such a route choice model from binary stated preference data. They show that a random coefficients model considerably improves the resulting log-likelihood, but that certain random parameters may have ranges that include both positive and negative signs – that is, a parameter may have a counter-intuitive value for some of its range.

We present a specification of **F** and **T** that accounts for path overlap in Section 3.3.2. We are unaware of any similar applications of hybrid logit or probit to large-scale route choice.

The Logit Kernel model suffers from the same computational difficulties as pure Multinomial Probit. Programs to estimate these types of models are widely available for use with Aptech

Systems GAUSS. For example, the code of Train, Revelt and Ruud (1999a and b) can be used to estimate models with diagonal **T** matrices.

2.1.13 Implicit Availability/Perception Logit

Cascetta and Papola (1998) introduced the Implicit Availability/Perception Logit model as a convenient way to incorporate awareness of paths into route choice modeling, without requiring an explicit choice set generation step. While IAP Logit uses a logarithmic correction term, as does C-Logit, care must be taken to distinguish the motivation for the correction. C-Logit uses the correction to adjust the MNL path share predictions, which the paths would receive if they were distinct. IAP Logit uses the correction term to decrease a path's share to reflect the possibility that travelers are unaware of that path, or unable to use it. The IAP Logit model gives the probability of traveler *n* choosing route *i*, $P_n(i)$, as

$$P_n(i) = \frac{e^{V_i + \ln \mu_n(i)}}{\sum_{j \in M} e^{V_j + \ln \mu_n(j)}}, \qquad (2\text{-}10)$$

where *M* is the *master choice set*, that is, the set of all possible routes. In this formulation, $\mu_n(i) = 1$ indicates that path *i* is available, while $\mu_n(i) = 0$ implies that either the path is unavailable or the traveler is unaware of it. That is, the limit of the log of zero is negative infinity, and the exponent of utility for an unavailable alternative is zero. The IAP Logit model could also be written as

$$P_n(i) = \frac{\mu_n(i) e^{V_i}}{\sum_{j \in M} \mu_n(j) e^{V_j}}.$$

When the analyst does not know $\mu_n(i)$ with certainty, it is treated as a random variable with expectation $\bar{\mu}_n(i)$. In this case, $\mu_n(i)$ may be replaced with its expected value in equation 2-10. However, Cascetta and Papola show a better approximation can be gained by using a second-order Taylor series expansion and by assuming the maximal variance for $\mu_n(i)$, which results from a Bernoulli distribution. The resulting model is

$$P_n(i) = \frac{\exp\left[V_i + \ln\overline{\mu}_n(i) - \frac{1-\overline{\mu}_n(i)}{2\overline{\mu}_n(i)}\right]}{\sum_{j\in M}\exp\left[V_j + \ln\overline{\mu}_n(j) - \frac{1-\overline{\mu}_n(j)}{2\overline{\mu}_n(j)}\right]}.$$

The authors then assume a binary logit specification for $\overline{\mu}_n(i)$:

$$\overline{\mu}_n(i) = \frac{1}{1+\exp\left(\sum_{k=1}^{K}\gamma_k Y_{ink}\right)},$$

where Y_{ink} is the k^{th} variable relating to the availability or perception (awareness) of alternative i for individual n, and γ_k are coefficients to be estimated.

Cascetta and Papola then describe a four mode (car, bus, metro and walking) choice model where the availability of car (Y_{car}) depends on car ownership, and the availability of the other modes depends on walking distance. The authors do not appear to have estimated an IAP Logit model using variables relating to awareness rather than availability.

2.1.14 Choice Set Generation and Route Choice

The implicit approach of IAP Logit to address what alternatives travelers consider in their choice sets can be contrasted with explicit choice set generation (CSG). Such an approach involves two stages: First, we determine which alternatives are considered by the traveler. Then, from among those alternatives, we determine which alternative is chosen. Random Utility Models are typically used in the second of these stages.

Noncompensatory choice set generation may be accomplished by using deterministic or stochastic rules. An example of a deterministic choice set rule is Elimination by Aspects (EBA), described by Tversky (1972). Under EBA, any alternative having an attribute value past some threshold (for example, travel time of more than an hour, or a transit trip requiring more than two transfers) is dropped from the choice set. Similarly, certain socioeconomic characteristics may eliminate some alternatives from being available. (For example, lack of a drivers' license or sufficient income to purchase a car will preclude driving alone.) Deterministic choice set rules often rely on heuristics and rules of thumb. Thresholds are usually calibrated by observing and minimizing the number of chosen alternatives eliminated by such rules.

Stochastic rules may also be used to establish which alternatives are likely to be in a traveler's choice set. This technique requires considering the full set of possible alternatives – that is, the master choice set, *M* – and is often called Probabilistic Choice Set Generation or Latent Choice Set Generation. Much of the following discussion follows from Gopinath (1995) and Ben-Akiva and Boccarà (1995). The general equation for the share or probability of an alternative, *i*, is

$$P_n(i) = \sum_{C \in G} P_n(i \mid C) P_n(C) \tag{2-11}$$

where *G* is the set of all possible choice sets, that is, the set of all non-empty subsets of *M*. If the availability (which may depend on awareness or other reasons, such as physical constraints or legal restrictions) of an alternative cannot be inferred from the availability of other alternatives, *G* will consist of $2^J - 1$ choice set elements, where *J* is the number of alternatives in the master choice set, *M*. That is, each alternative may be available or not (2^J possibilities), but since a choice must be possible, at least one alternative must always be available, and so the empty choice set is excluded from *G*.

Equation 2-11 may be seen by applying the so-called Chain Rule of Probability. Consider a very large contingency table where all alternatives are listed in rows, and choice sets are listed in columns. Some cells will be filled with structural zeros; that is, the alternative is not a member of the corresponding choice set. The product within the summation, $P_n(i \mid C) P_n(C)$ is the joint probability that an individual considers choice set *C* and chooses alternative *i*. We observe an alternative *i* being chosen, and so we are interested in its marginal probability.

It is difficult to observe an individual's choice set directly. Even the most intrusive surveillance equipment cannot observe the identity of unchosen alternatives. Travelers may be asked to list possible alternatives in surveys, but it is difficult to distinguish between alternatives that are unavailable and alternatives that are so unattractive the traveler fails to perceives them as available. Therefore, most choice set generation approaches treat an individual's choice set as a latent class. It is also possible for other latent variables (for example, the utility of an alternative, or an individual's time or cost sensitivity) to influence the availability of alternatives.

We have described possible specifications of the choice model, $P_n(i \mid C)$, in earlier sections. We must specify the choice set generation model, $P_n(C)$. Some techniques for modeling the choice set include captivity, random constraints and independent availability. Each is described briefly below.

Captivity models assume a simplification to the set G that the traveler either has all alternatives available, or is captive to the chosen alternative. That is,

$$G = \{\{1\},\{2\},\ldots,\{i\},\ldots,\{J\},\{1,2,\ldots,i,\ldots,J\}\}.$$

The probability that a traveler is captive to an alternative may be specified as a fixed proportion, or a function of socioeconomic characteristics, for example, income.

Such a captivity assumption makes it easy to estimate equation 2-11, as only two of the terms being summed are non-zero. However, this computational simplicity comes at the expense of placing severe restrictions on the interpretation of the model. For example, we may believe a traveler considers three to five alternative routes, which is neither being captive to a single route nor knowing all possible routes in the network.

A random constraints specification may be used to explicitly consider the factors that lead to an alternative being considered. For example, travelers may be willing to walk only a certain (individually varying) distance to transit. Owning a car may require meeting a certain income threshold. In general, we can write a random constraint as

$$A_{in}^{*} = 1 \Leftrightarrow h_{ink} \geq \nu_{ink} \quad \forall k \in K_i$$

where A_{in}^{*} is the latent availability of alternative *i*, that is, $A_{in}^{*} = 1$ if alternative *i* is available to or considered by individual *n*;

h_{ink} is a "criterion function" relating to alternative *i*, which may be a function of coefficients, socioeconomic characteristics, and attributes of the alternative;

ν_{ink} is a random threshold with zero mean; and

K_i is the set of constraints relevant to alternative *i*.

For example, if $d_{transit,n}$ is the distance individual *n* must walk to the nearest transit stop, we might write a constraint

$$A_{transit,n}^{*} = 1 \Leftrightarrow \beta d_{transit,n} - \alpha \geq \nu_{transit,n}$$

where β relates the scale of distance (perhaps meters or miles) to the scale of the random term $\nu_{transit,n}$, and α/β is therefore the average distance that all travelers are willing to walk to transit. Ben-Akiva and Boccarà (1995) provide other examples of random constraints.

The probability that a choice set, *C*, is the set considered by a traveler is therefore

$$P_n(C) = \frac{P_n\left(\left[A_{in}^{*} = 1\,\forall i \in C\right] \text{and} \left[A_{jn}^{*} = 0\,\forall j \in M \setminus C\right]\right)}{1 - P_n\left(A_{ln}^{*} = 0\,\forall l \in M\right)}$$

or

$$P_n(C) = \frac{P_n\left(\left[h_{ink} \geq \nu_{ink}\ \forall k \in K_i, \forall i \in C\right] \text{and} \left[h_{jnk} < \nu_{jnk}\ \text{for at least one}\, k \in K_j, \forall j \in M \setminus C\right]\right)}{1 - P_n\left(h_{lnk} < \nu_{lnk}\ \text{for at least one}\, k \in K_l, \forall l \in M\right)},$$

where $M \setminus C$ is the complement of $M \cap C$ in *M*, or colloquially, the "set subtraction" of elements of *C* from *M*. Note that the normalization in the denominator is needed because the choice set cannot be empty. The form of $P_n(C)$ depends upon the distributions assumed for v_{ink}, with logistic and probit models being the most common.

Not surprisingly, the independent availability model results by assuming that the alternative availability terms, A^*_{in}, are independent of each other, or in the case of random constraints, that the v_{ink} terms are independent. This allows the choice set model to be written using the multiplicative property of independent probabilities:

$$P_n(C) = \frac{\prod_{i \in C} P_n\left(A^*_{in} = 1\right) \prod_{j \in M \setminus C} P_n\left(A^*_{jn} = 0\right)}{1 - \prod_{l \in M} P_n\left(A^*_{ln} = 0\right)}$$

and

$$P_n(C) = \frac{\prod_{i \in C} P_n\left(h_{ink} \geq v_{ink} \ \forall k \in K_i\right) \prod_{j \in M \setminus C} P_n\left(h_{jnk} < v_{jnk} \text{ for at least one } k \in K_j\right)}{1 - \prod_{l \in M} P_n\left(h_{lnk} < v_{lnk} \text{ for at least one } k \in K_l\right)}.$$

Ben-Akiva and Boccarà (1995) show that a latent choice set generation model can have better fit than a simple MNL model. Even greater statistical efficiency can be gained if indicators of the choice set are used; however, this requires a more complicated model system than the one illustrated here.

2.1.14 Comparison of CSG and IAP Logit

It is useful to compare the assumptions and results of the IAP Logit model against an explicit choice-set approach using independent availability. Consider the very simple case of only two alternatives. Alternative 1 is assumed to be universally available, while the availability of alternative 2 is random. Initially consider $V_1 = V_2 = V$.

An IAP Logit model yields the probability of choosing alternative 1 as

$$P_n(1) = \frac{e^V}{e^V + e^{V+\ln \mu_n(2)}} = \frac{1}{1+\mu_n(2)}\,.$$

MNL with explicit choice set generation gives

$$\begin{aligned} P_n(1) &= P_n(1 \mid C = \{1\}) P_n(C = \{1\}) + P_n(1 \mid C = \{1,2\}) P_n(C = \{1,2\}) \\ &= (1)\left(1 - P\left(A_{2n}^* = 1\right)\right) + \left(\tfrac{1}{2}\right) P\left(A_{2n}^* = 1\right) = 1 - \frac{P\left(A_{2n}^* = 1\right)}{2} \end{aligned}\,.$$

Note that for MNL with explicit choice set generation, $P_n(1)$ is linear in $P\left(A_{2n}^* = 1\right)$, while for IAP Logit, the relationship is hyperbolic in $\mu_n(2)$.

For alternatives with different values of systematic utilities, we can show that under IAP Logit,

$$P_n(1) = \frac{1}{1+\mu_n(2) e^{V_2 - V_1}}\,,$$

while MNL with explicit choice set generation gives

$$P_n(1) = 1 - P\left(A_{2n}^* = 1\right) P_n(2 \mid C = \{1,2\}) = 1 - P\left(A_{2n}^* = 1\right)\left(\frac{1}{1+e^{V_1 - V_2}}\right).$$

We can therefore establish the equivalence between the parameters of IAP Logit and explicit choice set generation models as follows:

$$P_n(1) = \frac{1}{1+\mu_n(2) e^{V_2 - V_1}} = 1 - P\left(A_{2n}^* = 1\right)\left(\frac{1}{1+e^{V_1 - V_2}}\right).$$

Algebraic manipulation gives

$$\mu_n(2) = \frac{P\left(A_{2n}^* = 1\right) e^{V_1 - V_2}}{1 + e^{V_1 - V_2} - P\left(A_{2n}^* = 1\right)}\,.$$

Therefore, care must be taken not to represent the $\mu_n(i)$ term of IAP logit as the probability that alternative *i* is available, this is, $P\left(A_{2n}^{*}=1\right)$, but rather as a transformation involving such probabilities and the utilities of alternatives.

2.1.15 Summary of Models

Table 2-4 provides a summary of the route choice models discussed in section 2.1. Each of the models displayed assumes stochastic traveler behavior, with the exception of User Equilibrium.

With the exception of the Probit model, all other stochastic models are based on the Generalized Extreme Value (GEV) family of distributions. Latent variables and classes included in route choice models may be specified with any number of distributions; Gaussian, logistic, and lognormal distributions are common. The labeling technique is based on a theoretical model of how travelers may perceive different possible routes. That is, travelers become aware of a route to solve a particular objective or label. A physical route may correspond to more than one label, which requires a nested logit structure to address the correlation in awareness.

The C-Logit, Path-Size Logit, Cross-Nested Logit and Probit are distinguished from MNL by addressing path overlapping. For C-Logit and Path-Size Logit, an adjustment to expected utility is made. In the CNL and Probit models, overlap determines the correlation structure of the random error terms associated with each route.

IAP Logit, Elimination by Aspects and Probabilistic Choice Set Generation are means of considering availability or awareness of alternative routes. That is, without these techniques, the choice set must be assumed to be the same for all travelers, or to be exogenously given. EBA may be deterministic or stochastic, while the other methods are stochastic.

User Equilibrium, MNL and Labeling are among the easiest techniques to implement. Probit, Cross-Nested Logit and Probabilistic Choice Set Generation are among the most computationally demanding. For this reason, we observe that CNL and PCSG have not been implemented on networks of realistic metropolitan scale.

Model	Considers Overlap?	Requires Path Enumeration?	Author Associated with Theory	Can Be Estimated with "Canned" Software?	Can Be Applied with "Canned" Software?	Used with Large Networks?	Computational Effort Required
User Equilibrium	No	No	Wardrop	N/A	Yes	Yes	Low
SUE / MNL	No	No	Dial	Yes	Yes	Yes	Low
Labeling / NL	No	Yes	Ben-Akiva	Some Specs	No	Yes	Moderate
C-Logit	Yes	Yes	Cascetta	Yes	No	Yes	Moderate
Path Size Logit	Yes	Yes	Ben-Akiva	Some Specs	No	No	Moderate
Cross-Nested Logit	Yes	Yes	Prashker, Bekhor, Vovsha	No	No	No	High
Multinomial Probit	Yes	Yes	Daganzo	No	No	Yes	High
Logit Kernel	Yes	Yes	Ben-Akiva, Bolduc, McFadden, Train	No	No	No	Moderate to High
EBA & MNL	Can	Yes	Tversky	No	No	Yes	Moderate
CSG & MNL	Can	Yes	Ben-Akiva	No	No	No	High
IAP Logit	Can	Yes	Cascetta	Some Specs	Some Specs	No	Moderate
MNL w/ Latent Explanatory Variables	Can	Yes	Ben-Akiva	No	No	No	Moderate to High

Notes: "Canned" software for estimation includes Microsoft Excel, SAS and SPSS procedures, Dubin-Rivers SST, HCG ALOGIT, and Stratec HieLoW.

"Canned" software for applications include transportation packages such as Caliper TransCAD; UAG TP+, TRANPLAN and MINUTP; and INRO EMME/2.

Table 2-4. Comparison of Route Choice Models.

2.2 Transportation Information Studies

The previous section examined the various types of models that may be used to predict actions of travelers. This section turns to how models may be used to evaluate the impact of information systems on traveler behavior. Transportation professionals are interested in predicting how widespread the use of these information systems may be, how this information will impact travel patterns and whether the expense to collect the travel data disseminated by these systems is justified.

Much of the literature to date has focused on drivers' response (such as diversion while en route) to information about non-recurrent congestion in a dynamic context. While much of the benefit from ATIS may come from avoiding non-recurrent congestion, this thesis focuses on a more basic subject: the habitual (and therefore pre-trip) auto route choice decision in the context of the driver's limited awareness of alternative routes. En-route diversion is not explicitly considered. However, travelers' self-reported ability to make en-route diversions is one of the variables used to infer network knowledge. While this thesis focuses on static, long-run behavior, this section includes discussion of dynamic behavior for completeness.

Several studies have questioned whether, and under what conditions, providing travel information to individuals will result in improved performance of the transportation network. Arnott, de Palma and Lindsey (1991) show that under certain assumptions, imperfect information can lead to worse outcomes than no information at all. Mahmassani and Jayakrishnan (1990) assume homogenous driver route choice preferences and calculate that system-optimal penetration rates of ATIS are less than 100 percent. That is, their simulations suggest that optimal travel conditions depend on denying information about travel conditions to a fraction of travelers! Ben-Akiva, de Palma and Kaysi (1991) describe *overreaction* (too many drivers divert in response to a congestion report, creating worse delays on an alternative facility) and *concentration* (reliance on ATIS leads travelers to select from a smaller subset of alternative routes) as possible undesirable outcomes of travelers having expanded access to information about travel conditions. The authors also warn of *oversaturation*. That is, drivers who receive information need that information to be concise and well-organized; otherwise drivers may be overloaded with the mental task of classifying and responding to travel information while also concentrating on the physical task of driving. If drivers are oversaturated by incoming travel information, a degradation in safety – which could possibly have a greater social cost than that of traffic congestion – would likely result.

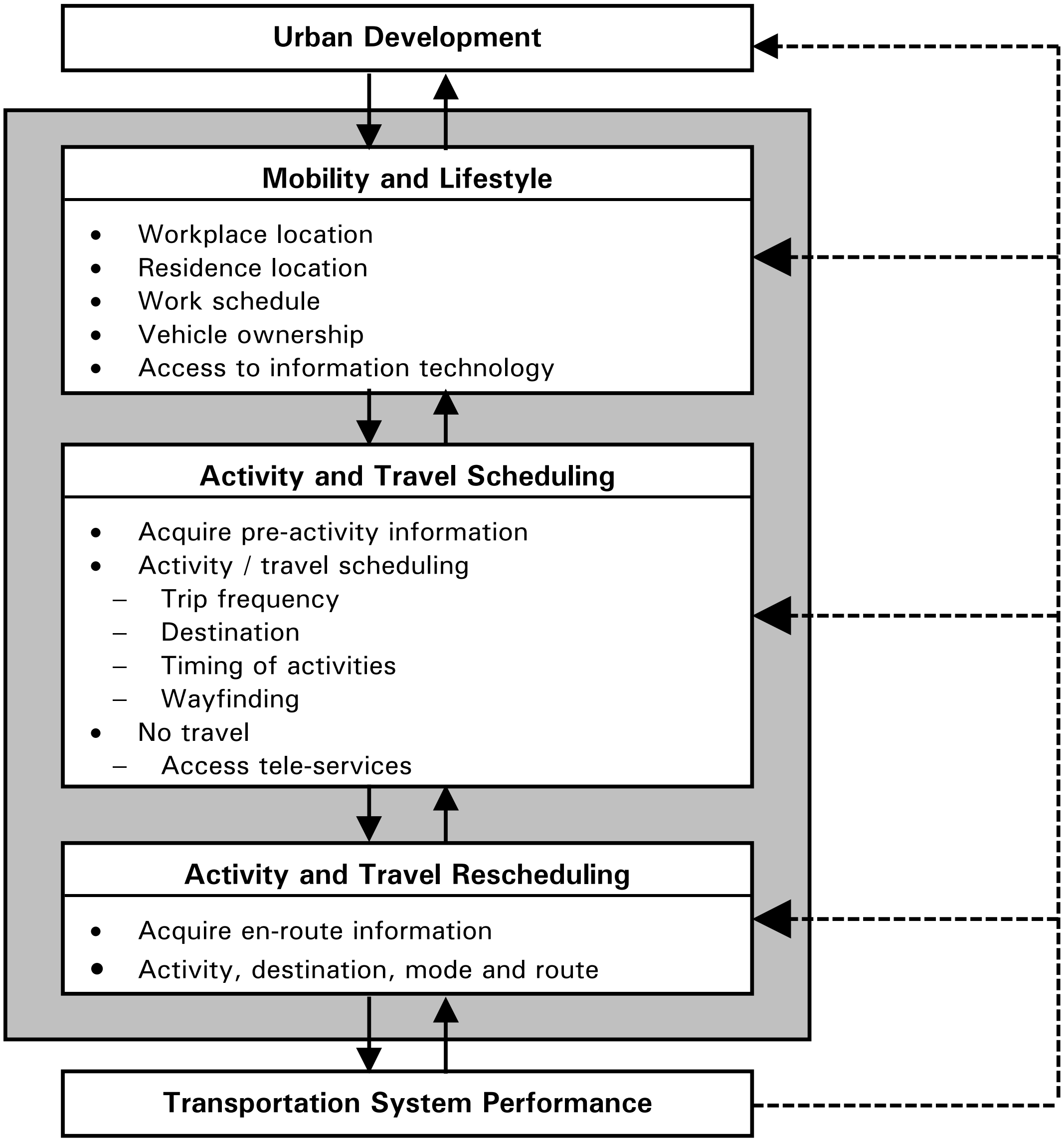
Other authors are less cautious about the benefits of ATIS, because the models that predict such dire outcomes may be subject to severely limiting assumptions. Delvert (1997) argues that models of travel behavior in response to ATIS must address heterogeneity in behavior and mental processing ability. Gopinath (1995) demonstrates that considerably different model forecasts result when heterogeneity of travelers' value of time (time-cost trade-off) is considered. It is reasonable to conclude that the proposed models considering heterogeneity of network ability would produce significantly different results than conventional models. Models that reflect differences in network ability and knowledge would offer guidance for designing conventional and advanced traveler information systems should benefits accrue primarily to specific populations. The transportation disadvantaged – who may be over-represented by the elderly, persons with disabilities or those with limited incomes – may have unique information needs. For example, studies such as that by Golledge, Jacobson, Kitchin and Blades (1999), which describes theories of spatial ability of blind people, will be useful in developing ATIS for that population.

Using information and traveling are complex behavioral processes, which involve many types of decisions and possible outcomes. Ben-Akiva, Bowman and Gopinath (1996) present a hierarchy of choices consisting of awareness, access, use, travel response and learning. This hierarchy is reproduced in Figure 2-4.

The impact of information on travel may be classified in several ways: (1) Travelers can only choose from options of which they are aware, so information and knowledge affects choice set generation. (2) Awareness of alternatives also depends on the information sources consulted, so models of information use must be integrated with models of perception formation and spatial choice. (3) Travelers' decisions are made according to travelers' perceptions of alternative attributes, which are formed both through experience and from information travelers acquire. This research places special emphasis on the first and third points.

Only limited research has examined the third point, how travelers form perceptions of alternatives from their experience and travel reports. Most studies that address perception of attributes, such as Jha, Madanat and Peeta (1996); Kaysi (1992); and Iida, Akiyama and Uchida (1992) assume travelers update their estimates of attributes such as travel time in a mathematical manner analogous to Bayesian statistical estimation. The Bayesian updating technique therefore assumes travelers have sufficient working memory to form new perceptions by averaging new experiences with previous perceptions. Lotan (1997) uses a fuzzy numerical representation to model travelers perceptions of travel times. These fuzzy perceptions are used by a heuristic decision model, which consists of rules such as "if travel time on Route A is more than X minutes, don't use it," and "always use Route A if the travel time on it is less than Y minutes." Lotan compares the results of this heuristic Approximate Reasoning for Transportation (ART) model against multinomial logit, but does not include the fuzzy perceptions in the logit models considered. That is, the difference in predicted shares cannot be distinguished between the use of fuzzy perceptions and the use of heuristic decision rules.

Likewise, very few studies have addressed the first impact of information, the issue of travelers' ability to navigate through their surroundings. Stern and Leiser (1988) describe the spatial knowledge of residents and professional drivers in Beer Sheva, Israel. Deakin (1997) examined the extent to which Southern California commuters became familiar with alternative routes when their primary freeway route was closed for repair after the 1994 Northridge Earthquake. Khattak and Khattak (1998) estimate a two-stage model of awareness of alternative routes, and en-route diversion.


Urban Development
Mobility and Lifestyle
• Workplace location
• Residence location
• Work schedule
• Vehicle ownership
• Access to information technology
Activity and Travel Scheduling
• Acquire pre-activity information
• Activity / travel scheduling
– Trip frequency
– Destination
– Timing of activities
– Wayfinding
• No travel
– Access tele-services
Activity and Travel Rescheduling
• Acquire en-route information
• Activity, destination, mode and route
Transportation System Performance

Source: Ben-Akiva, Bowman and Gopinath (1996)

Figure 2-4. Hierarchy of Choices in Transportation Networks.

2.2.1 Access to Information Technology

Studies of ATIS adoption often consider two stages, awareness of the information system, and purchase of, or access to, information technologies and services. Just as awareness may refer to travelers' knowledge and perceptions of the transportation network, use of guidance depends on knowledge and perceptions of its various sources, including both the conventional and emerging media mentioned earlier. Awareness of information sources obviously influences which are consulted. Travelers compare received messages with experienced travel times to form perceptions of the quality and relevance of guidance sources. Typical studies of awareness include Polydoropoulou, Ben-Akiva and Gopinath (1997), which develop latent variable models of awareness, trial use and repeat use, and Walker (1994), which illustrates how the *information acceleration* technique may be used to simulate travelers becoming aware of ATIS products and services.

Access is the action or set of actions that allow a traveler to receive guidance. In some cases, this involves the purchase of hardware, such as in-vehicle route guidance devices, or a computer to view traffic web sites from home. Specialized hardware manufacturers may refer to their market share as the penetration or adoption rate. Parish (1994) considers the time trajectory of ATIS penetration by making comparisons to related technologies, electronic tolling and traffic management (ETTM) – which includes electronic toll collection (ETC) – mobile telephones, and automatic vehicle identification or location (AVI or AVL). Subscriptions offered by travel information content providers are another type of access. Access studies attempt to establish which attributes are most desired by various segments of consumers, how ATIS offerings compare to existing information sources, and travelers' willingness to pay for access. For some technologies, such as variable message signs (VMS), it is not meaningful to discuss the traveler's access decision.

2.2.2 Information Use and Traveler Response Models

Usage models consider each instance when travelers seek and receive messages. Important considerations in this step include travelers' perceptions of both information sources and current traffic conditions, and constraints on receiving messages (e.g., can messages only be received at home? in the vehicle?). Information use studies may be classified by the time at which information is accessed and therefore the travel decisions of interest, and by the data collection techniques used. Researchers may use traditional survey techniques (e.g., fill-out form or telephone interview) and ask about past situations in which travelers encountered unexpected congestion or travel information (revealed preferences or RP data), or about hypothetical situations (stated preferences or SP data). In other studies, researchers develop elaborate computerized travel simulators to collect SP data under conditions designed to make subjects more cognizant of the commuting task.

Abdel-Aty, Kitamura and Jovanis (1996) provide an example of a study of ATIS for pre-trip decision-making. They describe an idealized system that provides transit travel time information to homes by TV, radio or computer network. The authors use an SP approach to examine the effect of travel information on mode choice by asking respondents how likely they would be to use transit at least one day a week if the ATIS reported various travel times. Responses were recorded on a 10-point likert scale. The authors estimated an ordered probit mode choice model of transit use conditional on information received. To examine the impact of information would seem to require comparison of coefficients of a route choice model given conventional information, but the authors do not appear to undertake this analysis. (However, a separate section of the survey examined what characteristics travelers desired in an ATIS.)

Many studies consider both pre-trip and en-route decisions. In some cases, researchers are interested in the affects of ATIS on all types of travel decisions. For example, the reports by Khattak, Polydoropoulou and Ben-Akiva (1996) and Polydoropoulou, Ben-Akiva, Khattak and Lauprête (1996) describe two concurrent surveys. The first report examines a survey where travelers were asked about their response to a hypothetical pre-trip information source. Response to an in-vehicle ATIS is the topic of the second report and survey. The information dissemination medium may also allow travel information to be obtained both before and during travel. As an example of such a medium, Khattak, Schofer and Koppelman (1995) first consider radio traffic reports, and examine how commuters respond to those reports. They then ask commuters how they would respond to a hypothetical ATIS with similar access characteristics.

Khattak, Polydoropoulou and Ben-Akiva (1996) asked Golden Gate Bridge commuters to recall when they had last learned of delays on their usual route before leaving home. Responses were classified as changing route, leaving earlier, leaving later, changing mode, both changing route and leaving earlier, canceling the trip, and making no change. Then respondents were asked to consider an ATIS that resembled a TV screen with text messages. Message types examined were

- *qualitative information*, such as "unexpected delays on your usual route;"
- *quantitative information*, which includes current estimates of delays on the usual route and travel times on the next best alternative route;
- *predictive information*, which provides instantaneous estimates of the delay on the usual route and predictions of delay durations for vehicles arriving at the delay 15 and 30 minutes in the future; and
- *prescriptive information*, in which the device compares travel times internally and may advise taking an alternative route or mode.

The authors used combined RP/SP techniques to estimate models of driver response to ATIS messages. They concluded that drivers exhibit some inertia for using their habitual route. Those who are unfamiliar with alternative routes or modes are particularly unwilling to divert. However, accurate quantitative information may be able to overcome this behavioral inertia. Further, the commuters were generally willing to comply with advice from a prescriptive ATIS.

Polydoropoulou, Ben-Akiva, Khattak and Lauprête (1996), survey another group of Golden Gate Bridge commuters to model their response to observations and reports of upcoming congestion. First, respondents were asked to recall the last time they became aware of unexpected congestion while driving on their usual route. For this situation, they were also asked to record whether they diverted to another route, and various data about the delay (e.g., its duration, whether it was caused by an incident or weather). Then respondents were asked to consider an in-vehicle device giving various types of text messages, similar to the pre-trip ATIS described above.

The authors use a combined RP/SP estimation approach to model route switching. The results indicate that travelers have considerable resistance to using alternative routes, especially for home-to-work trips. Drivers are more likely to divert to another route when they first learn of a delay through the radio, although the authors hypothesize it may be too late for drivers to divert when they can observe the delay visually. Bay Area commuters are just as likely to divert when they become aware of accidents as they are for other causes of congestion; however, they are less likely to divert during bad weather, as alternative routes are equally slow. Prescriptive information greatly increases travelers' diversion probabilities, although similar diversion rates are attainable by providing real-time quantitative or predictive information about travel times on usual and alternative routes. The authors suggest that drivers would prefer to receive travel time information and make their own decisions.

Khattak, Schofer and Koppelman (1995) examined a survey of travelers who commute by auto to downtown Chicago during the morning peak. About 70 percent of these travelers use radio traffic reports – the larger fraction while driving – and generally rate the reports positively. Downtown commuters felt the radio traffic reports were relevant (since they cover primarily radial freeways), but desired more accuracy and timeliness. The authors determined that many commuters were either willing to change both route and departure time or unwilling to change either aspect of their travel. Khattak, Schofer and Koppelman estimate a bivariate ordinal probit model of willingness to change route or departure time given traffic information. That is, the two dependent variables are five-point likert scale responses indicating that travelers may "Strongly Agree" to "Strongly Disagree" that they will change aspects of their travel. The authors determined that route switching is more likely when the usual route is regularly congested, the driver is in the habit of choosing a route after departing from home, the usual route is fairly new (and little choice inertia has been established), or when the driver is male or wealthier. Departure time changes are more likely when the usual route has reliable travel times, and when the alternative route takes considerably longer.

Abdel-Aty, Vaughn, Kitamura, Jovanis and Mannering (1994) consider a 1992 telephone survey of LA commuters, which contains information about whether the commuter follows the same route each morning or uses multiple routes, whether the traveler receives pre-trip or en-route information, and the frequency of route changes. The authors estimate three models: (1) a bivariate probit model of whether commuters access pre-trip information and whether they use multiple routes, (2) a bivariate probit model of whether commuters access en-route information and use multiple routes, and (3) a negative binomial model of the frequency of route changes given pre-trip or en-route information use.

The authors observe that women tend to listen to pre-trip information more, while more men receive en-route information. Freeway users who perceive heavy congestion on their route are also more likely to receive pre-trip information. Only 15 percent of the commuters use more than one route to work; these people tend to have higher incomes than the remainder of survey participants. The authors estimate a negative correlation between pre-trip information use and use of multiple routes, which suggests that commuters may be more likely to use pre-trip information to adjust departure time. The correlation between en-route information use and use of multiple routes was insignificant.

Pre-trip and en-route information use have a positive influence on the number of route changes. Those who make longer trips and use pre-trip information are likely to make more route changes. Among en-route information users, car-poolers, those who perceive heavy traffic, and those who perceive the information to be accurate are likely to make the greatest number of route changes.

Liu and Mahmassani (1998) use a series of networked computers to collect simulated commuting decisions from 45 University of Texas at Austin faculty and staff members for an experimental period representing five workday mornings. Decisions of each of the participants affect the experienced network travel times. The authors use this data to estimate a multivariate probit model where the 24 dependent variables are four days worth of decisions to change departure time, to change route before leaving home, and to change route at each of four en-route decision points. Their results suggest that departure time decisions are correlated with those of previous days, and with the pre-trip choice of route. However, no significant correlation was detected between the departure time decision and en-route diversions made later that day. Travelers were more likely to change departure time or route when their current choice would cause them to arrive late. Also, drivers exhibited some inertia in route choice, requiring a travel time savings of at least about a minute on the new route.

Adler, Recker and McNally (1993) describe a simulator (FASTCARS) for examining drivers' pre-trip and en-route choices in an arbitrary network with various information sources. Static signs (such as freeway exit signs) provide information about names and locations of roadways. Traffic congestion information is available from VMS or Highway Advisory Radio (HAR). Travelers also have the option of using an in-vehicle navigation device to direct them on the shortest path under current congestion levels. However, use of HAR involves a fixed cost, while drivers using the navigation device encounter a fixed cost and a per-minute charge. Conventional roadside signs and VMS are free.

At the beginning of each simulated trip, participants may set their own goals by establishing weights for minimizing schedule delay, travel time, number of stop lights encountered, distance and number of road changes. Drivers may also select a route and departure time before leaving home. Drivers may access information, change route or change their goals while they travel. At the end of the trip, drivers can see their performance and may further adjust their goal.

Adler and McNally (1994) use the FASTCARS simulator to examine how travelers become aware of the fictional city of Terrapin over a period of up to 10 days. Subjects recruited from university students, professors and administrative staff were instructed to imagine leaving at 6:30 p.m. for a special event that starts at 7:30 p.m. The drivers may belong to one of three familiarity levels (low, medium and high), which affects what information is available when they consult a map. Low-familiarity drivers saw only a subset of freeways in Terrapin, and only distances were displayed. Medium-familiarity drivers saw all the freeways and a subset of arterials. These drivers could observe distances and speeds. The high-familiarity drivers were able to observe all links in the network, and obtain distances, speeds and accident frequencies. During the first trial, drivers were randomly assigned to be low-familiarity drivers with 90 percent probability, or medium-familiarity drivers. Drivers advanced to the next level of familiarity after successfully completing three trials. Since the computerized map was intended as a way of exogenously providing the mental map in long-term memory, there was no penalty for accessing it, unlike for accessing ATIS. Drivers of all familiarity levels could see the upcoming mile in the simulator window, and all drivers could access ATIS.

The authors conclude that more familiar travelers are less likely to consult information. They observe that unfamiliar drivers who access HAR or the in-vehicle navigator did not score as well as familiar drivers. However, note that drivers are assessed a penalty for accessing HAR or navigation information. The authors did not appear to investigate unfamiliar driver performance under different penalty schemes, representing different ATIS pricing structures.

2.2.3 Studies of ATIS Benefits

This section addresses the evaluation of alternative ATIS configurations. The benefits of ATIS largely involve changes in travelers' decisions resulting in improved network performance, and travelers' perceptions of better traveling conditions. This section draws from the discussion published in Ben-Akiva, Bottom and Ramming (2000).

First we describe the distribution of various types of benefits to information users, other travelers and citizens, and to the system or society as a whole. Then we examine various criteria that information providers may consider. Finally, we provide archetypal examples of the various techniques that have been employed by researchers to evaluate ATIS.

ATIS users receive direct benefits from traffic messages. The most obvious of these are travel time savings and a less difficult driving environment that results from not having to drive under severely congested conditions or fumble with maps in unfamiliar areas, for instance. Travelers may also find ATIS attractive because the messages provided will result in them having greater confidence in the travel times expected. With these improvements in reliability, travelers will be able to better plan their daily activities. When congestion is unavoidable, messages from an ATIS may allow travelers to call ahead (if a mobile phone is available) to notify colleagues or family members of the delay. ATIS may also prompt travelers to postpone or select alternative destinations of discretionary trips. Less familiar travelers may also benefit from an ATIS assisting them in learning new routes for habitual trips. Finally, guidebook systems with destination information may allow travelers to benefit through learning new destinations at which to complete their sustenance and recreation activities.

Externalities result in non-users receiving benefits from ATIS. If traffic flows more smoothly, both ATIS users and non-users benefit. Studies such as Mahmassani and Jayakrishnan (1991) verify the intuition that ATIS users receive a disproportionate share of the benefits. Residents living near heavily-used transportation facilities may benefit from reduced noise or pollution that accompanies smoother-flowing traffic. Finally, because commercial vehicles may use ATIS, private citizens may benefit through the reduced costs of delivering goods being reflected in retail price reductions.

The system-wide or societal benefits are of course the sum of all user and non-user benefits. These benefits can be classified as travel time savings, environmental improvements (such as reduced noise or airborne emissions), and improved economic efficiency. Brand (1998) warns that it is possible for travel time reliability to result in travelers making more or longer trips, and therefore vehicle-miles or vehicle-hours of travel (VMT or VHT) provides an incomplete representation of ATIS benefits.

Different types of actors or agents may be involved in providing ATIS. Various levels of government may participate in the collection or dissemination of network data. Private firms may also be involved in the manufacture of guidance equipment or in the resale of travel data as an added-value service. In some settings, public-private partnerships may result in a division of responsibilities. For example, in the TravInfo project for California’s Bay Area, public agencies (primarily CalTrans and the many transit operators) collect network data for a uniform database. This database is then sold to private concerns, who may process and simplify that data for transmission to consumers.

Each type of actor may have different motives. Obviously, private firms will be motivated to earn profits by selling travel data or ATIS equipment to travelers. Since travelers desire to reduce their travel time or improve the reliability of travel, private firms will have incentives to provide messages according to the user optimality principle. Public sector entities, in contrast, are charged with multiple and sometimes disparate objectives. For example, a transport department may wish to improve travel times, but while maintaining environmental quality. Such public agencies may therefore consider both user optimal and system optimal objectives.

ATIS evaluation studies may be classified by type of benefit presented above – travel time, reliability, and environment, for example – concerning user or system benefits, public or private objectives, and dimension of traveler response considered in the two previous sections. Additionally, it may be useful to distinguish studies by the type of methodological treatment adopted. We propose three groupings of analytical techniques, although particular studies may adopt one or more such techniques. Theoretical or Analytical studies typically examine small networks (sometimes only two parallel links) and develop closed-form solutions of ATIS impacts from assumed volume-delay and traveler behavior relationships. Simulation studies examine more realistic sized networks using given volume-delay functions and behavioral responses to predict the travel changes resulting from ATIS. Simulations may resemble conventional transportation models by considering continuous flows on links (*meso-scale*), may model individual or groups (*packets*) of drivers at the *micro-scale*, or may adopt different scales for different types of models (e.g., estimate the state of travel demand at the meso-scale, but examine auto maneuvers at the micro-scale to estimate the congested travel times). The final methodology type, an Econometric study, is concerned with estimating parameters of behavioral relationships and making inferences from the values of these parameters (e.g., comparing the coefficients on subscription cost and information quality in an access choice model to establish the value of incremental information).

Arnott, de Palma and Lindsey (1991) is an example of an Analytical study. The authors examine the conditions under which the provision of information may result in worsening traffic conditions. For a network consisting of two routes (each a single link), low probabilities of capacity reductions and low quality information may conspire to worsen system-wide travel costs.

One common approach adopted by simulation studies involves a multi-class assignment where unguided users follow stochastic user optimal principles (that is, unguided users have same-variance travel time perception errors) while guided travelers follow (deterministic) user optimal paths. More sophisticated simulation studies use microscopic dynamic traffic assignment and locally-derived behavioral parameters. Examples of simulators are DynaMIT (see Ben-Akiva, Bierlaire, Koutsopoulos and Mishalani, 1998), DYNASMART-X (Mahmassani and Hawas, 1998; Hawas, Mahmassani, Ziliaskopoulos, Ghang and Peeta, 1997), METROPOLIS (de Palma and Marchal, 1998) and TRANSIMS (Nagel, Barrett and Rickert 1996).

Econometric studies generally focus on one or two aspects of traveler response to ATIS and estimate behavioral parameters from observed behavior (revealed preferences) in areas where early deployments of ATIS are available, intentions (stated preferences) under hypothetical situations described in experimental or survey settings, or combinations of both RP and SP data. Conclusions may be made directly from values of behavioral parameters estimated, or by using these parameters in simulation studies. Examples of econometric studies include Polydoropoulou, Ben-Akiva and Gopinath (1997), Khattak, Polydoropoulou and Ben-Akiva (1996), Polydoropoulou, Ben-Akiva, Khattak and Lauprête (1996) and Caplice and Mahmassani (1992). The next section describes the geographical theories that may guide specification of econometric models.

2.3 Cognition and Decision

Gärling (1998) and Gärling and Friman (1998) summarize theories that relate cognition to the decision-making process. Experience and spatial knowledge are stored and recalled from long-term memory, which is essentially permanent and has a high capacity. However, decision-making occurs in working memory, which has a limited capacity. That is, facts related to options under consideration (attributes of alternatives) must be recalled from long-term memory and brought to working memory for consideration.

There are two prevailing theories of spatial cognition: One model uses a biological metaphor and asserts that neural structure mimics the hierarchical spatial structure of the environment being represented in memory. The other model uses a computational metaphor — people store rules of how to respond to combinations of stimuli in much the same way that computer programs contain myriad if-then statements.

O'Neill (1991) describes a biological model of spatial knowledge based on Kaplan's (1976) cognitive map theory. Some neurons represent "nodes" or individual places, while other neurons represent neighborhoods, areas, cities or regions. Neurons are physically connected to other neurons representing nearby places. Neurons are also connected to other neurons representing larger and smaller spatial units in the mental hierarchy. When a person constructs a path between two places, the neurons representing the origin and destination are stimulated. If those neurons are connected — that is, the origin and destination are close — the path is recalled. Otherwise, neurons in increasing levels of the spatial hierarchy are stimulated until a complete path is connected and recalled.

In contrast, Computational Process Models (see Smith, Pellegrino and Golledge, 1982) hypothesize that spatial information is stored as a list of rules and properties of objects. In this model, a place is described by an *identity* or *cue* (for example, "Massachusetts Institute of Technology"), a generic *class* or *function* (a university), and a location (at Massachusetts Avenue and Memorial Drive in Cambridge, MA). Gale, Golledge, Pellegrino and Doherty (1990) state that navigation ability arises by associating the knowledge of identity of places with knowledge of how to get between places.

Because humans have limited information processing capacities, they often apply heuristic rules to simplify decision-making tasks. Such heuristics may include rules for eliminating alternatives that fail to have an acceptable value of a certain attribute. Such a decision protocol is called Elimination by Aspects (EBA), which was formalized by Tversky (1972). As an example of decision protocols used when compensatory evaluation is too taxing, Hirtle and Gärling (1992) describe three heuristics travelers use for the goal of minimizing distance traveled during a multiple-destination tour.

After heuristics reduce the choice set to a workable number of alternatives, people may switch to using a compensatory decision rule, such as that suggested by economic utility maximization.

The Random Utility Theory that is the basis for the discrete choice models discussed in section 2.1 assumes that people are able to consider all the alternatives' attributes in working memory. Attribute values must be accessible in working memory for people to make compensatory judgements, that is, to assess whether one alternative possesses enough more of one attribute to make up for the fact that it lacks some of another attribute in comparison to other alternatives. It is important to recall that economists do not assume that people calculate a utility value for each alternative, but that the results of such a process are sufficiently similar to what consumers are observed to do.

2.4 Critique of Utility Theory

The discussion now turns to the validity of an important assumption in most models of individual travel and spatial choice — that of utility maximization. There has been much debate in the behavioral science and economics communities over the validity of some of the assumptions of utility theory, which is the basis for most models of travel demand. Among a wide literature, Kahneman and Tversky (1984), and Tversky (1977) illustrate some cognitive "anomalies" that appear to violate rational utility maximization. These anomalies may relate to the context and framing of the experimental task presented to study participants, or to individuals' limitations in information processing capability.

McFadden (1997) describes three components of economic rationality assumptions: (1) rationality in perceiving attributes of the choice problem, (2) rationality or stability in individual preferences, and (3) rationality of the decision-making process. He concludes that the volume of behavioral evidence definitively disproves only rationality of perceptions. There is no conclusive evidence whether preferences may vary randomly and in response to changes in situations, or whether no underlying preferences can be deduced from choice data. Therefore, additional research is necessary to establish rationality of preferences or tastes. Svenson (1990) describes how compensatory rules apply when the decision-maker is sufficiently involved in the choice task. The level of decision-making intensity depends on a person's engagement with the choice problem, and the context and consequences of the choice. At the lowest level of engagement, people simply match the current situation to past experiences, and select the option that was chosen previously, without regard to the attributes or outcomes of the alternatives. At the second level of decision-making, a person recalls the attractiveness of each option, and groups alternatives into "good" and "bad" choices. The third level of decision-making involves the tradeoffs among multiple attributes, which is the assumption of utility theory. The fourth level of decision-making is similar to problem-solving and includes the possibility of generating new alternatives.

Prospect Theory (Kahneman and Tversky, 1979; Tversky and Kahneman, 1992; Tversky and Fox, 1995) drops the more restrictive assumptions of traditional utility theory by allowing sensitivity to the status quo and non-linear responses to gains and losses with respect to the initial conditions. Prospect Theory models have been more successful at explaining empirical observations of choice than traditional utility theory. Camerer (1989) concludes that the adaptations to utility theory made by Prospect Theory are sufficient to explain a number of "anomalous" empirical observations. Likewise, we adopt the approach of using random utility theory and explicitly incorporating cognitive effects.

2.5 Geographic Theory and Results

To test cognitive and behavioral theories, geographers often conduct experiments in which a relatively small sample (say 40 to 100 individuals) is asked to perform certain navigation or orientation tasks under controlled circumstances (e.g., limited exposure to a new environment, whether the environment was experienced directly or through multimedia reproduction). These experiments give insight into how people may mentally organize spatial data, and how spatial data is acquired. Some results typical of current geographic theory are presented below.

Freundschuh (1992) summarizes the different regimes for storing spatial data, which represent various degrees to which spatial information is integrated:

- Persons with *landmark knowledge*, sometimes called *declarative knowledge* or *geographical facts*, are able to recall the characteristics (cue and function) and location of a place.

- The second kind of geographical knowledge is *route knowledge*, where people are able to link landmarks with directions for getting from place to place. Because route knowledge includes directions for navigation, it is sometimes called *procedural knowledge*.

- The third regime is *map knowledge*, *survey knowledge* or *configurational knowledge*. Persons with this ability know the interrelationship of places and routes with each other (that is, *topology*). Map knowledge often includes information about distances and angles between features (*metrics*).

Freundschuh devised an experiment to test whether characteristics of the spatial environment (regular grid streets versus more “organic” layouts typical of suburban cul de sacs) and experience influence attainment of spatial knowledge. He was unable to obtain statistically significant results suggesting that greater length of residence or a regular spatial environment would lead to greater map knowledge. Instead, he concluded that the organization of spatial information is an individual characteristic, and that models assuming homogenous spatial knowledge are unrealistic. Freundschuh’s results did not provide conclusive evidence on whether these spatial knowledge schemes are "levels" that people "pass through" as they gain navigation experience; however, these regimes remain useful for classifying different individuals according to processes by which they may make wayfinding decisions.

Gale, Golledge, Pellegrino and Doherty (1990) investigated the ability of nine- to twelve-year-old children to develop route knowledge sufficient for navigation in an unfamiliar suburban neighborhood. One group of children learned the neighborhood through field experience, while a second group learned solely from a videotape of a walk through the neighborhood. The field learning group was able to navigate more effectively, while the video group performed better at declarative tasks (e.g., identifying whether a house was located in the subject neighborhood). The authors concluded that route knowledge is quite parsimonious, with navigation information concentrated at intersections, which are decision points. The field group "filtered out" much of the landmark information not relevant to the navigation task.

Gärling, Böök, Lingberg & Arce (1990) examined the question of whether and how elevation is encoded in cognitive maps. A convenience sample of first- and third-year university students were asked to report whether they had visited various landmarks, and to give relative and quantitative estimates of elevation for the landmarks. The results indicate that even the less-experienced group had gained knowledge of elevations. Another experiment established that the time necessary to recall or estimate the elevation between two places is not proportional to the distance between them. This result suggests that elevation is encoded with landmark knowledge, rather than requiring a route between the two points to be mentally retraced. This result has implications that the mental encoding of another spatial variable — travel time — may not be immediately apparent. Instead, alternative hypotheses about encoding of travel time should be examined. Encoding and recall are two processes in the perception of travel times.

The result of this literature review has been to identify a consistent and justifiable set of behavioral assumptions from which quantitative models of travel and spatial decisions can be developed. The literature articulates theories that different people have different methods of encoding spatial data, and that knowledge of physical space is organized in identifiable ways. In the next chapter, we describe how to incorporate these theories into mathematical models of travel and spatial choice.

2.6 Synthesis

This chapter presented several transportation models used by practitioners and researchers alike. The assumptions of these models, particularly those regarding travelers' awareness of routes and perception of route attributes were examined, and limitations were discussed. Special emphasis was given to transportation models to examine the impact of information-providing technologies. We also examined current biological, behavioral and geographical theories about how people make decisions in a spatial environment. One important conclusion that may be drawn from this overview is that travelers are quite heterogeneous in their behavior – including their sensitivities to travel times and costs; inclination to avoid difficult driving maneuvers, unsafe neighborhoods or unattractive roadways; and awareness of possible alternative routes. Accounting for this heterogeneity requires more complex modeling techniques. For instance, the different mental organization structures for spatial knowledge (landmark, map and route) should form a convenient scheme for classifying travelers into distinct behavioral groups. Making this classification scheme into an operational mathematical model is the topic of Chapter 3.

3 Methodology

In this chapter, we present a general methodology that may be used to model travelers' network knowledge and resulting route choices in a variety of contexts. There are three components of this methodology that are discussed in this chapter. The first is modeling network knowledge from travelers' attitudinal data. Another component is generating possible alternative routes that resemble those a traveler would use. Finally, the network knowledge model and choice set generation process is used in the development of a route choice model, which may take any of the forms discussed in Chapter 2, such as MNL, Path-Size Logit, C-Logit, Cross-Nested Logit, IAP Logit, Logit Kernel or hybrids of these.

In Section 3.1, we provide an overall modeling framework, present a refinement of the Path Size formulation, and explore perceptions of travel time. Section 3.2 describes the plan for coding and testing alternative path generation algorithms. The discussion of Section 3.3 concentrates on explanatory variables for network knowledge and path utilities, and the details of applying Logit Kernel to route choice.

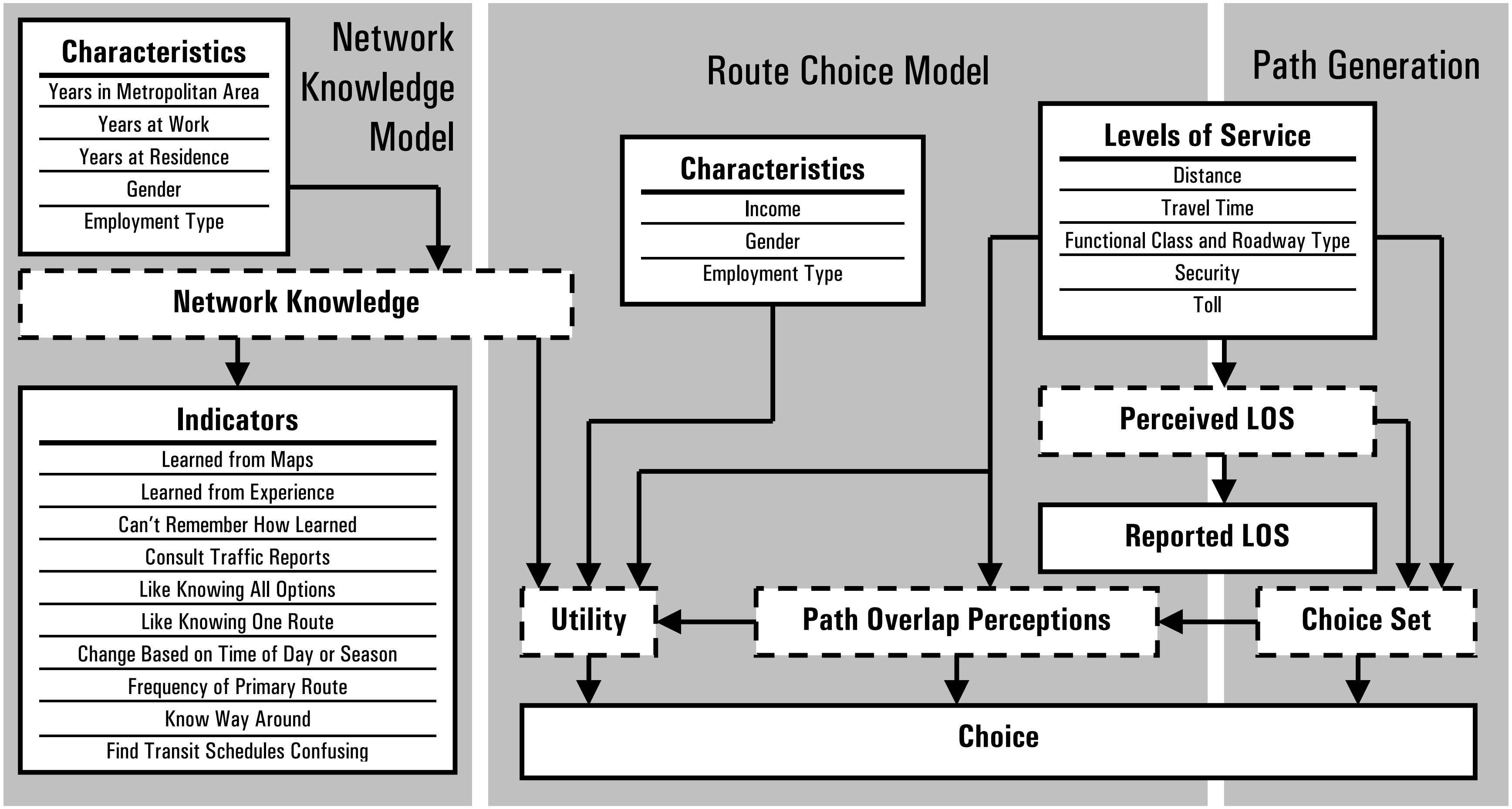
3.1 Behavioral Hypotheses

In this section, we discuss our hypotheses regarding how traveler characteristics may relate to network knowledge, and how network knowledge may affect route choice. Section 3.1.1 presents a general model framework for studying route choice in the context of knowledge and information. In section 3.1.2, we discuss a general framework for calculating Path Size, and how these mathematical formulations can express travelers' perceptions of overlapping paths. The final section (3.1.3) describes how travelers' perceptions of their travel time (as reported on surveys) may not match the time calculated for that trip from network variables. That variation may be used to guide making random draws for path generation, as described in Section 3.2.6.

3.1.1 Modeling Framework

To better model travelers' perceptions of alternative routes and their preferences for routes requires three broad techniques. First, we wish to describe travelers' network knowledge from their socioeconomic characteristics and attitudes toward navigation. Then we wish to generate a set of alternative routes that incorporates travelers' perceptions of network variables and their preference for those variables. That is, the route generation procedure should be capable of generating routes similar to those travelers are observed to make. (As this discussion progresses, we will refine the intuitive concept of "similar" into a measurable quantity called "coverage.") Finally, we wish to model route choice in such a way that addresses both travelers' trade-offs of route attributes and the degree to which travelers consider a route. This modeling framework is presented graphically in Figure 3-1. These three techniques or modules are represented by a grey box in the background. White boxes in the foreground represent the types of variables of interest in each module. Examples of broad class of variables – traveler characteristics, indicators of network knowledge, and levels of service – are given. The figure uses the convention of a path diagram; that is, causality or influence is shown by arrows. Latent variables – or variables that are hard to measure accurately – are shown with dashed lines.

The leftmost module of Figure 3-1 deals with relationships to network knowledge. Recall that network ability refers to the mental capabilities and organizational schemes a person has available for remembering the transportation network, while network knowledge refers to the information about the network the person has been able to experience and learn. Ideally, our modeling framework would allow us to consider both network ability and network knowledge. For example, network knowledge may be expressed as a person's choice set and values of perceived attributes of those routes in the choice set. However, data collection techniques may



Notes: Latent variables are shown in dashed boxes.

Figure 3-1. **Schematic of the Route Choice Process.**

limit our ability to distinguish between the concepts of knowledge and ability. Network ability may be genetic. Therefore, it may be more difficult to correlate socioeconomic data available from conventional surveying techniques with network ability. Questions eliciting network ability may involve the travelers' preferences for knowing multiple routes or making spatial decisions. However, it may be difficult for people to think about their navigational preferences without considering their current situation, so responses may also reflect the network knowledge travelers have acquired. For simplicity, we develop models of network knowledge directly from socioeconomic variables and responses to attitudinal surveys.

An unfamiliar traveler may acquire network knowledge from many sources. First, he or she may get routes from others – friends, relatives, colleagues, taxi drivers or information lines. Learning where the landmarks are will be required to follow the route instructions. Therefore we are interested in characteristics relating to the travelers' situation. For example, does the traveler have regular contact with people from whom he or she may learn new routes? How long has the traveler been able to acquire knowledge about the city? Are there cultural factors such as gender or race that may affect how a person learns routes? We are also interested in the travelers' assessment of his or her navigation skills. Are there preferred information sources or media? Does the traveler change routes frequently, implying knowledge of a larger number of options? What is the travelers' attitude to decision-making in general? These form some of the attitudinal indicators of network knowledge.

During the learning process, the traveler may also look at maps, which may speed integration of data about the new city into map knowledge, if he or she possesses map ability. At this point, it is interesting to consider why some routes are easily learned or learned first. We hypothesize that route properties may facilitate learning. Such properties may include being a freeway, high capacity roadway, highlighted on a map, or numbered and signed. A route that uses few streets, makes few turns and can be described concisely may also be more easily learned, or at least conveyed quickly by more knowledgeable travelers. We describe such attributes of a route that help it be more easily or quickly learned as increasing a route's *prominence*.

Finally, a traveler with map ability may learn shortcuts and explore smaller arterials. That traveler might adopt a route using a higher proportion of these less-used roads to avoid congestion on more popular (and more prominent) routes. ITS proponents may assert that one potential benefit of ATIS is making people aware of less prominent routes, and therefore better utilize the existing capacity of less prominent routes.

While it may be difficult, it is important to distinguish between utility and prominence. Utility refers to those factors causing a route to be chosen. That is, travelers are assumed to conduct a rational search process and find the routes with the greatest utility first. However, utility is not the whole story. Some routes may be more easily learned, but less effective. We refer to the capability of a route to be easily learned as prominence. This concept is similar to the attributes of the built environment that geographers consider when examining which places are more or less likely to be remembered or used as a landmark. Prominence of a route may be related to physical characteristics, operational or functional characteristics, informational characteristics, and factors that interact with traveler characteristics. Since travelers are searching for their preferred routes, it is reasonable to expect prominence and utility to be correlated.

The relationship between prominence and utility is also clouded by the traditional two-step procedure for modeling route choice: First, possible alternative routes are generated to form the choice set. This is shown in the rightmost module of Figure 3-1. Then the probability a given route is chosen from a specified choice set is calculated. This is shown as the middle module, labeled the “Route Choice Model.”

These two procedures may correspond to non-compensatory and compensatory decision rules. Of course, the distinction need not be so strict. Recall that the motivation for the IAP Logit model is to add a correction to the utility term (in the choice modeling step) to compensate for including routes the traveler is not aware of during the path generation step.

The two-step methodology we present has the advantage that by explicitly specifying the set of available routes, we can examine possible selection criteria, and reduce computational time by not generating unrealistic routes. With a finite, known choice set, we can apply theoretically-based corrections for route overlapping, such as the Path Size correction, which is the topic of the next section.

3.1.2 Overlapping Paths and a General Path Size

As described in Section 2.1.10, the introduction of a Path Size term can help overcome the problem that the MNL model assumes independence between paths, which is not true when paths overlap. Many different Path Size formulations are possible. Therefore, it is helpful to have some guidance in selecting a Path Size formulation. These criteria may be based on theoretical, empirical or practical considerations.

Theoretical considerations relate to the interpretation of Path Size. For example, Path Size should not be affected by network coding, such as splitting a link in half at some intermediate point, or recoding a two-way link as two one-way links. Distinct paths (that is, paths that share no links with other paths) should have a size of one, and overlapping paths should have a size less than one. Path Size is dependent on the choice set definition; however, it is desirable for it to be robust to the inclusion of questionable paths.

Of course, it may be possible to construct examples where Path Size Logit gives counter-intuitive results. To the extent that these examples contain pathological routes – that is, routes no traveler would be expected to use – these examples do not represent a sufficient argument against the PSL approach. The choice probability of such routes under other model types – for example, CNL or LK – may also be counterintuitive. Instead, a reasonable choice set generation procedure is needed to exclude such pathological routes.

Empirical considerations mean that we want a good model fit. Statistics like rho-bar-squared and the t-statistic of the Path Size coefficient can help guide our selection of formulation, and any parameters in that formulation.

An example of a practical consideration is that Path Sizes should be easy to calculate with reasonable computational effort. The Link-Path incidence matrix is expected as an input to any Path Size formulation (or to any Commonality Factor formulation, for that matter).

The simple, conceptual formula for the size of a path, PS, is

$$PS_{in} = \sum_{a \in \Gamma_i} \frac{l_a}{L_i} LSC_{ain} \tag{3-1}$$

The first term in the summation, l_a / L_i, is a weight by which link-specific terms are summed to form the Path Size. The second term of Equation 3-1 may be thought of as a link size contribution (LSC_{ai}). That is, links may also be thought of as having a “size” of one, which is allocated among the paths using that link. The size of a path is then the sum of these “link sizes” weighted by the amount the link contributes to the overall path. For a distinct link or series of links, the path accrues the full size contribution from those link(s). The total path size also depends on the link size contributions accrued from other links in the path. When more than one path share a link, the “link size” of one is split among the paths.

To illustrate how the PSL choice probabilities compare against other model types, we present the overlapping path problem of Figure 2-3 again below. The three paths all have the same length, T, and Paths 1 and 2 overlap for a length of $T - d$. Clearly, MNL predicts equal shares for the three paths – one third each – which is correct only when $d = T$.

The choice probabilities for the overlapping paths (that is, Paths 1 or 2) are presented graphically in Figure 3-2. The horizontal axis is the fraction of T that d represents. That is, at zero, $d = 0$, so Paths 1 and 2 are two separate “labels” or “names” for the same physical path. In this case, we

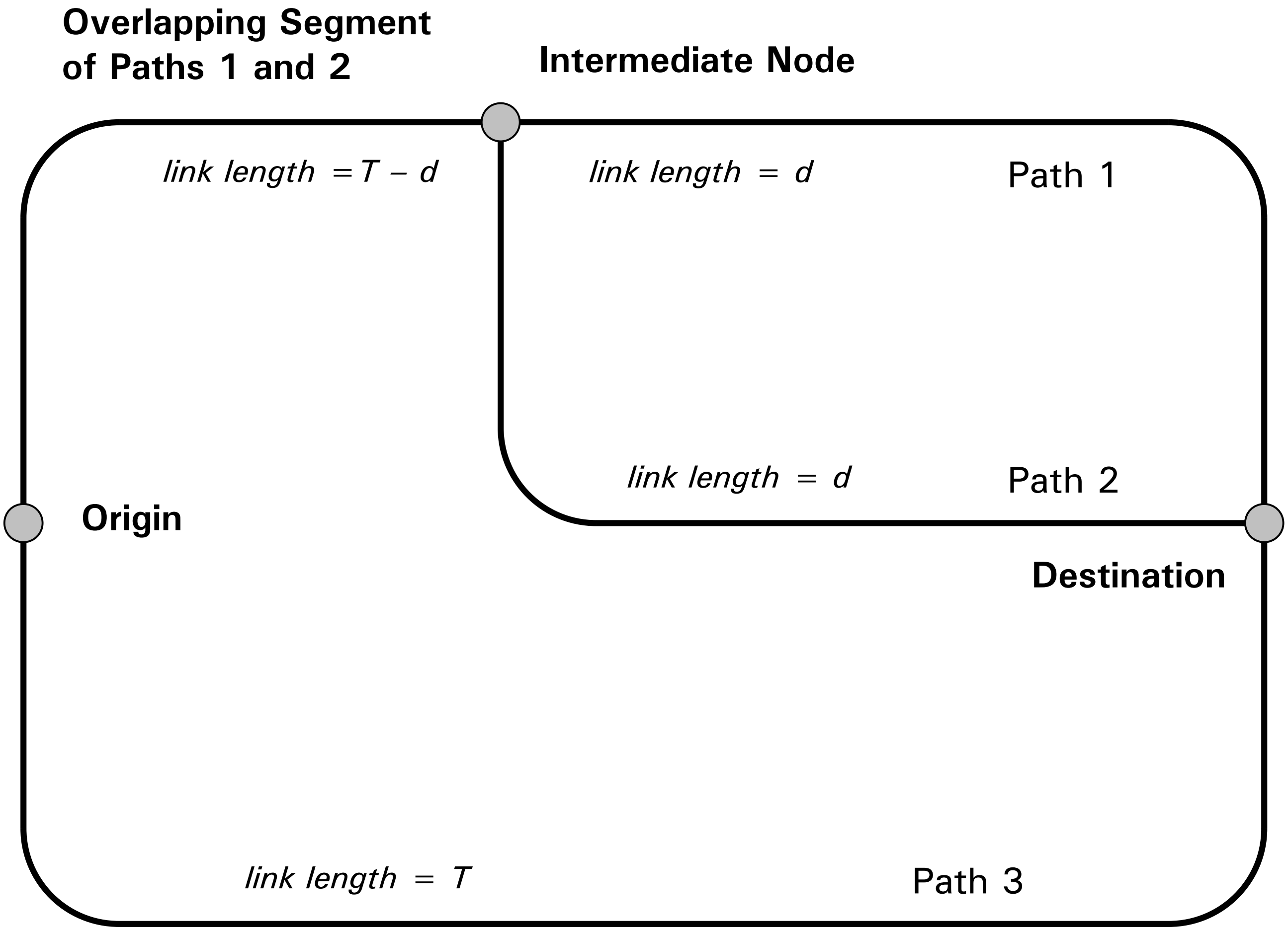
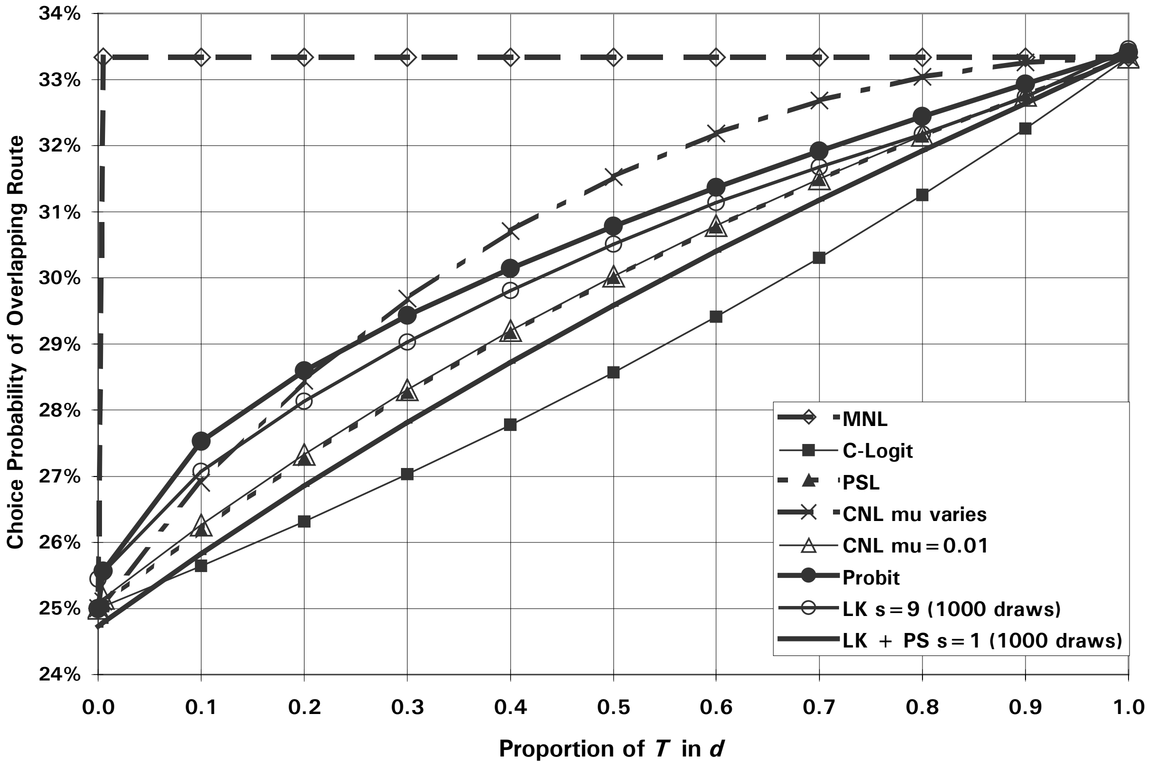


Figure 2-3. The Overlapping Path Problem (Duplicate).



Notes: C-Logit uses the commonality factor of Equation 2-6 with $\beta_0 = 1$. "CNL mu varies" uses the specification of μ given by Equation 2-9. The LK parameter σ = s as shown. The same random draws are used to calculate the LK probability for each data point.

Figure 3-2. Choice Probabilities for the Overlapping Path Network.

expect the choice probabilities for Paths 1 and 2 to be 25 percent each, and for Path 3 – the other physical path – to have a choice probability of 50 percent. One on the horizontal axis of Figure 3-2 corresponds to $d = T$.

Note that the MNL choice probability is flat at 33 percent since MNL is sensitive only to the relative length of the paths (which are equal) and not to overlapping. At zero, we assume the network is redefined, and therefore there is a "jump" in probability to 25 percent. Instead of a jump, we expect a smooth curve for the choice probability in response to changes in the length of the overlapping link. C-Logit, PSL, CNL and Probit all have the correct choice probabilities for the extreme cases ($d/T = 0$ and 1), and smooth changes in probability. The LK and LK + PS curves are close to the theoretical values for the extreme cases, which reflects some of the error inherent in using simulation to calculate these choice probabilities. (The specification of these models and the use of simulation is discussed later, in Section 3.3.2.)

The curve for "CNL mu varies" (that is, mu is given by Equation 2-9) is concave downward and the closest to the flat MNL curve. The Probit curve is also concave downward, and the LK curve follows the Probit curve closely at a slightly lower choice probability. This is as expected, since the LK model is specified with assumptions similar to those for Probit, but the LK model has an additional Gumbel error term. Note that the PSL curve is similar to the curve for a Cross-Nested model with $\mu = 0.01$. This curve is slightly concave downward. Next, the LK + PS curve follows close to the PSL curve. In contrast, the C-Logit curve – based on the commonality factor of Equation 2-6 – is concave upwards. We have no intuition regarding which of these curves is most reasonable; instead, we must test the models' fit empirically.

It can be demonstrated that when one of the overlapping paths is substantially longer than another, the method for allocating the LSC must be carefully chosen to avoid counter-intuitive route choice probabilities. For example, an earlier formulation developed to address the issue of longer overlapping paths was as follows:

$$PS_{in} = \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \left(\frac{1}{N^*_{an}} \right), \tag{3-2}$$

where $N^*_{an} = \sum_{j \in C_n} \delta_{aj} \frac{L^*_{C_n,a}}{L_j}$, and $L^*_{C_n,a} = Min_{j' \in C_n,\ \delta_{aj'}=1} L_{j'}$, that is, the length of the shortest path in C_n using link a. In this formulation, the count of paths among which a link's "size" is split is represented by N^*_{an}. Paths which are among the shortest paths through the subject link count as "full" paths in N^*_{an}, while longer paths make a reduced contribution. In this way, arbitrarily long paths – which would likely not be considered by travelers – do not reduce the size of other, more reasonable paths that use the same link.

However, note that in this formulation, the LSC term does not add up to one when summed across all paths using a link. This is most easily seen by noting that both the shortest and the longest path receive an LSC of $1/N^*_{an}$, while the longest path does not count as a "full" path in the calculation of N^*_{an}. It can be shown that this formulation produces counter-intuitive results for certain networks.

We can adapt the formulation of Equation 3-2 so that the LSC sums to one by changing the numerator for longer paths. The Path-Size formulation then becomes

$$PS_{in} = \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \left(\frac{L^*_{C_n,a} / L_i}{N^*_{an}} \right) = \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \left(\frac{L^*_{C_n,a} / L_i}{\sum_{j \in C_n} \delta_{aj} \frac{L^*_{C_n,a}}{L_j}} \right).$$

However, note that $L^*_{C_n,a}$ is constant with respect to the summation over alternative paths j. The $L^*_{C_n,a}$ terms cancel, and the formulation may be written as

$$PS_{in} = \sum_{a\in\Gamma_i} \frac{l_a}{L_i}\left(\frac{1/L_i}{\sum_{j\in C_n}\delta_{aj}\frac{1}{L_j}}\right) = \sum_{a\in\Gamma_i} \frac{l_a}{L_i}\left(\frac{1}{\sum_{j\in C_n}\delta_{aj}\frac{L_i}{L_j}}\right).$$

As will be shown below, this formulation may still produce counter-intuitive forecasts, but it does give insight into a more general formulation:

$$PS_{in} = \sum_{a\in\Gamma_i}\left(\frac{l_a}{L_i}\right)\frac{1}{\sum_{j\in C_n}\frac{G(L_i;\gamma)}{G(L_j;\gamma)}\delta_{aj}} = \sum_{a\in\Gamma_i}\left(\frac{l_a}{L_i}\right)\frac{1}{N^*_{an}}$$

where $G(\bullet)$ is a function with parameter γ. Note that this formulation satisfies the condition that

$$\sum_{i\in C_n}\delta_{ai}LSC_{ain} = 1$$

for all real links a. One example uses an exponential formulation:

$$PS_{in} = \sum_{a\in\Gamma_i}\left(\frac{l_a}{L_i}\right)\frac{1}{\sum_{j\in C_n}\frac{L_i^{\gamma}}{L_j^{\gamma}}\delta_{aj}} \tag{3-3}$$

Note that $\gamma = 0$ corresponds to the Ben-Akiva and Bierlaire (1998) definition shown in Section 2.1.10.

The problem of selecting values of γ can be illustrated with the network shown in Figure 3-3. Note that Paths 1 and 2 have the same length, 10. Path 1 is distinct, while Path 2 shares a link (link b) with Path 3, which is 2 units longer. Table 3-1 presents the choice probabilities for the three paths of Figure 3-3 predicted by the MNL model, and the exponential PSL model using

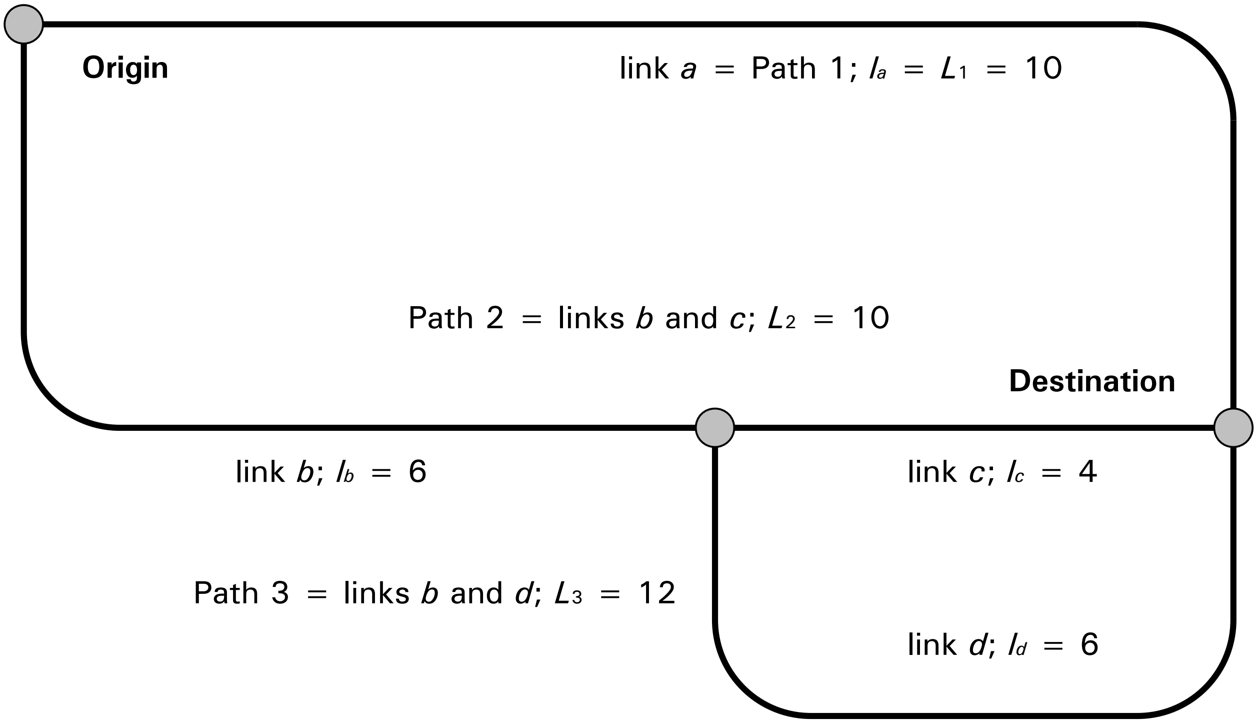

Origin
link a = Path 1; l_a = L_1 = 10
Path 2 = links b and c; L_2 = 10
Destination
link b; l_b = 6
link c; l_c = 4
Path 3 = links b and d; L_3 = 12
link d; l_d = 6

Figure 3-3. Network for Exponential Path Size Logit Example

Multinomial Logit		Exponential Path-Size Logit						
		$\gamma =$	0	1	2	4	11	∞
$P_n(1)$	46.8%	$P_n(1)$	55.5%	54.8%	54.1%	52.8%	49.9%	48.4%
$P_n(2)$	46.8%	$P_n(2)$	38.9%	39.8%	40.8%	42.5%	46.3%	48.4%
$P_n(3)$	6.3%	$P_n(3)$	5.6%	5.4%	5.2%	4.7%	3.8%	3.3%
		PS_{2n}	0.700	0.727	0.754	0.805	0.929	1
		PS_{3n}	0.750	0.727	0.705	0.663	0.559	0.5
$P_n(2)/P_n(2 \text{ or } 3)$	88.1%	LSC_{b2}	0.5	0.545	0.590	0.675	0.881	1
$P_n(2)/P_n(2 \text{ or } 3)$	11.9%	LSC_{b3}	0.5	0.455	0.410	0.325	0.119	0

Note: Totals may not add to unity due to rounding.

Table 3-1. Choice Probabilities for EPSL Example.

different values of gamma. The value of the coefficient on path length is assumed to be –1 for the example presented.

We hypothesize that the chosen $G(\bullet)$ function must split the link size contributions more severely than MNL would split path shares, or counter-intuitive predictions will result.

3.1.3 Travel Time Perception

As a final refinement, it is important to note that the travel times available to the analyst may not match the perceived travel times reported by survey respondents. Neither set of travel times may correspond to reality. Travelers may bias their reported travel time based upon the difficulty of the driving task: for example, heavy traffic or frequent stops may result in an over-estimate of trip time. Nor are the analysts' times without error. Travel times may be collected by vehicle detection equipment, which are subject to error, or estimated by a network model. Recall that network models are typically "calibrated" for a particular base year, which may not correspond to the survey year. The calibrated assignment may also represent "average" travel conditions rather than those of a particular day. Further, the omission of certain roads from the network means that traffic volumes that should be on the omitted road may get added to the volumes of an included link. No calibration data set is perfect, and modelers generally concentrate on matching volumes rather than times during the calibration stage. Recall that the relationship between volumes and times in planning networks is typically the BPR function, which may not correspond to empirical observations of traffic flow characteristics.

The relationship between the engineering estimates of travel times used in path generation and route choice modeling, and the perceived travel times reported by survey respondents is examined further in Section 4.3.

3.2 Choice Set Generation Plan

This section describes the mechanics of the route choice set generation stage. The first two sections deal with the measures by which we know we have developed a suitable route generation methodology. Section 3.2.1 discusses the notion of coverage, that is, the measure of how closely the algorithmically-generated path matches an observed path. Section 3.2.2 discusses computational concerns. The following sections review the design variables for some of the path generation algorithms discussed in section 2.1. Specifically, section 3.2.3 discusses calibration of the link penalty method. Section 3.2.4 describes the assumptions used in applying the link elimination method. Assumptions to make the simulation method operational are summarized in section 3.2.5

3.2.1 Coverage as an Objective

Ideally we would like to have computational algorithms that are capable of selecting the exact path that any traveler would. Of course, humans exhibit complex behavior, and algorithms cannot perfectly predict the routes all travelers would choose. To compare these imperfect algorithms, we develop criteria that relate how closely a generated route matches the observed route. First, we define some notation:

$n = 1 \ldots N$ indexes observations,
$i, j = 1 \ldots J_n$ indexes paths,
$k = 1 \ldots K$ indexes link variables, $\mathbf{X}$,
$a = 1 \ldots M$ indexes links, and
$r = 1 \ldots R$ indexes algorithms (“r” for “route”).

The most intuitive measure is how much two paths overlap. We define L_{nr} = the distance that the path generated by algorithm r overlaps with the observed path for individual n. Because travelers may have a large range of separation between their origins and destinations, it may be more useful to express this measure as a percentage, which we will call the *overlap percent*:

$Overlap\ Percent_{nr} = L_{nr}\ /\ L_n$,

where L_n = the length of the observed path.

Of course, if the overlap is 100 percent, the algorithm is consistent with the behavior used by that traveler. We may say that we have *replicated* the path for individual *n* with algorithm *r*. Since it is unlikely that any algorithm or combination of algorithms will replicate all survey routes, we need to develop some other objective functions. These include

$$\underset{r}{Maximize}\sum_{n=1}^{N} OverlapPercent_{nr} \text{ , and}$$

$$\underset{r}{Maximize}\sum_{n=1}^{N} 1\left(OverlapPercent_{nr} \geq \theta\right), \tag{3-4}$$

where θ is some chosen threshold and 1(•) equals one when its argument is true and zero when its argument is false. We can also define the Overlap Percent in terms of other link variables.

Recall that the objective of having good coverage from the path generation procedure is to insure we don't introduce bias in the choice model. For model estimation, maximizing overlap is even a more restrictive criterion than necessary. If the survey and generated route are close in "attribute space," we do not introduce bias in the estimation process. Consider a grid network for example. If all the blocks have similar travel times, it is not necessary to match the particular blocks a driver uses, but only the total path skims. Example objective functions based on these criteria include

$$\underset{r}{Maximize}\sum_{n=1}^{N}\sum_{k=1}^{K}\frac{X_{nrk}}{X_{nk}} \text{ , and}$$

$$\underset{r}{Maximize}\sum_{n=1}^{N}\sqrt{\sum_{k=1}^{K}\left(X_{rk} - X_{nk}\right)^2} \text{ ,}$$

where X_{nk}, X_{rk} and X_{nrk} are the path skims of link variable X_k for the observed route, the route generated by algorithm *r*, and their overlap, respectively.

For the evaluation presented in Section 5.1, we use a criterion of the form of Equation 3-4. One reason for this choice of coverage measure is that we wish to exclude observations with poor overlap so that they do not bias the route choice model estimation results. Examining overlap on an observation-by-observation basis also allows the opportunity to develop algorithms that attempt to replicate the behavior of those observations with poor overlap from other algorithms. Further, we choose the more strict criterion of link-by-link matching because we do not have many reliable reported path attributes from which an "attribute space" criterion could be calculated. Also note that an overlap criterion results in a collection of algorithms that would be useful for guidance generation as well as model estimation, because the links of generated paths would be similar to what a traveler would select.

It is possible to improve the above objectives by simply adding more and more algorithms until all survey paths are replicated. Of course, this would take additional computational resources, which is the topic of the next section. When developing a set of algorithms as the preferred methodology, we will examine the incremental contribution of one algorithm with respect to others. That is, an algorithm will be dropped from the "preferred set" of algorithms if it does not increase the coverage objective over the combined coverage of the remaining preferred algorithms.

3.2.2 Computational Concerns

Running time and disk space are our primary computational concerns. We found that disk storage was not a constraint, as only the algorithms that required long run times would also use much disk space. This follows from the observation that algorithms may write temporary files during their execution. The longer the run, the more disk space used by temporary files. For some algorithms, we divided the computational task into single OD pairs, while other algorithms operated on the whole data set.

Another computational concern is ease of implementation by practitioners. We wished to use software and techniques that would be available to typical MPOs. We are unaware of any common modeling program (e.g., Caliper TransCAD, INRO EMME/2, UAG TP+) that implements an exact solution to the K-Shortest Path problem, so we eliminated this algorithm from further consideration. Instead, we consider link penalty and link elimination heuristics, which can be performed by programming a macro to make repeated shortest-path calls, with appropriate network modifications between each shortest-path call. These heuristics are the topic of the next two sections.

3.2.3 Link Penalty Methods

Link penalty algorithms are described in section 2.1.7. The design variables of this algorithm are the initial impedance variable to use, how much to increase impedance at each iteration, and the number of unique paths to generate. We use estimated time as the impedance variable, and build up to 40 paths. The impedance increment is more tricky to calibrate, as the example network in Figure 3-4 and the sample link penalty runs described in Table 3-2 illustrate. If the parameter has too low a value (as in Table 3-2a), the algorithm is not computationally efficient, as the same path is identified over and over. If the value is too high (as in Table 3-2c), longer paths may be generated before paths that are more similar to the shortest path. Since computational resources may limit the total number of paths generated, it is desirable to have the most "realistic" paths generated first. We use an increment of 3 percent for origins that are very close to MIT, 5 percent for the most distant origins, and 4 percent for the remainder.

3.2.4 Link Elimination Methods

The design parameters of link elimination methods are the impedance variable, the number of links to eliminate at a time, and the number of paths from which to eliminate links. Note that the number of paths generated will depend on the number of links in the shortest path and on network topology. Estimated time is used as the impedance variable (as with the link penalty methods), and we eliminate only one link per iteration from the (first, most) shortest path. With these parameters, the link elimination algorithm produced as few as two and as many as 49 unique paths.

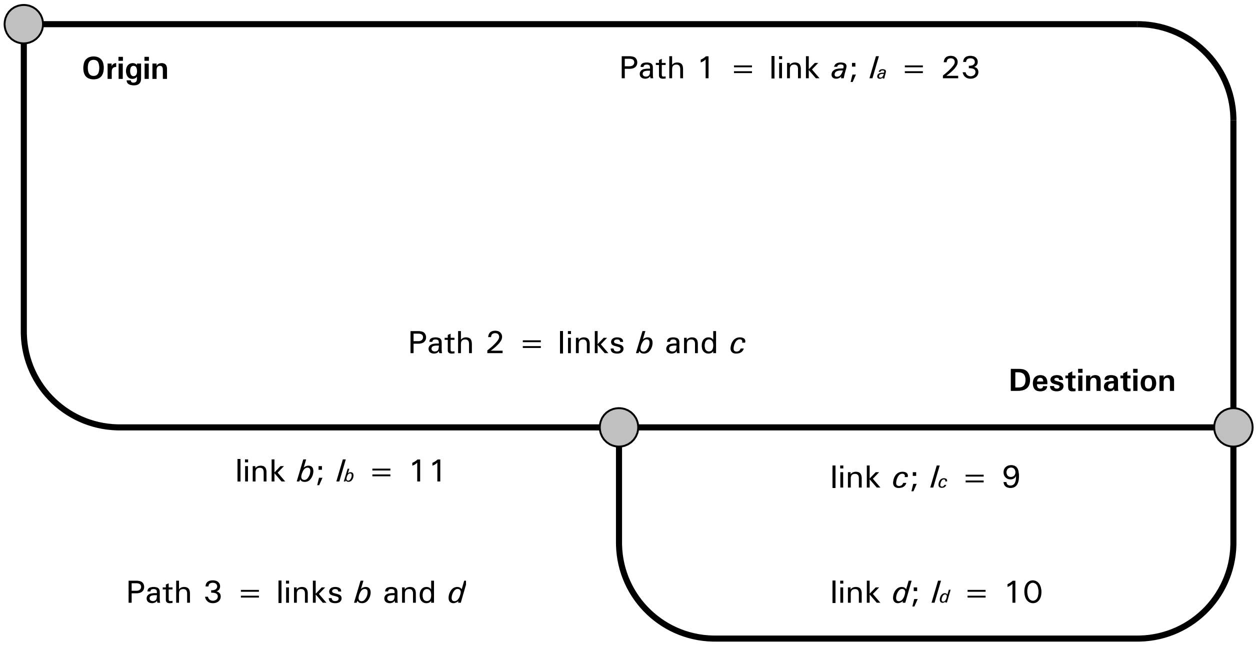

Origin
Path 1 = link *a*; l_a = 23
Path 2 = links *b* and *c*
Destination
link *b*; l_b = 11
link *c*; l_c = 9
Path 3 = links *b* and *d*
link *d*; l_d = 10

Figure 3-4 Example Network Illustrating Calibration of the Link Penalty Algorithm.

	Current Link Costs				Current Path Costs			Shortest Path	True Cost
Iteration	***a***	***b***	***c***	***d***	**1**	**2**	**3**		
1	23	11	9	10	23	20	21	2	20
2	23	11.3	9.3	10	23	20.6	21.3	2	20
3	23	11.7	9.5	10	23	21.2	21.7	2	20
4	23	12.0	9.8	10	23	21.8	22.0	2	20
5	23	12.3	10.1	10	23	22.4	22.3	3	21
6	23	12.7	10.1	10.3	23	22.7	23.0	2	20
7	23	13.0	10.4	10.3	23	23.3	23.3	1	23

a. Incrementing at 3 percent of original link impedance each iteration.

	Current Link Costs				Current Path Costs			Shortest Path	True Cost
Iteration	***a***	***b***	***c***	***d***	**1**	**2**	**3**		
1	23	11	9	10	23	20	21	2	20
2	23	11.7	9.5	10	23	21.2	21.7	2	20
3	23	12.3	10.1	10	23	22.4	22.3	3	21
4	23	13.0	10.1	10.6	23	23.1	23.6	1	23

b. Incrementing at 6 percent of original link impedance each iteration.

	Current Link Costs				Current Path Costs			Shortest Path	True Cost
Iteration	***a***	***b***	***c***	***d***	**1**	**2**	**3**		
1	23	11	9	10	23	20	21	2	20
2	23	12.1	9.9	10	23	22	22.2	2	20
3	23	13.2	10.8	10	23	24	23.2	1	23
4	25.3	13.2	10.8	10	25.3	24	23.2	3	21

c. Incrementing at 10 percent of original link impedance each iteration.

Note: Link impedances are rounded to the nearest one-tenth for simplicity in presentation.

Table 3-2. Example of Calibrating the Link Penalty Algorithm.

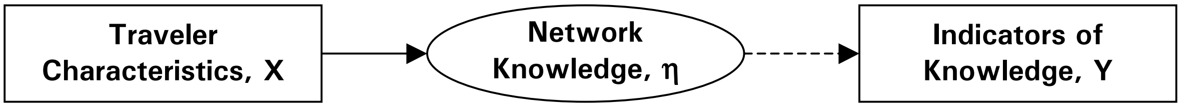
3.2.5 Simulation Methods

Simulation methods produce alternative feasible paths by drawing impedances from different probability distributions. The distribution type (for example, Gaussian, Gumbel, Poisson), distribution parameters, number of draws and the seed of the pseudo-random number generator are design variables. We use a Gaussian distribution with a mean and standard deviation from the model of travel time perception described in Section 3.1.3. We also examine increasing the standard deviation. We choose to make 48 draws from each distribution, as this was estimated to take roughly the same computational time as the link elimination and link penalty algorithms.

3.3 Model Specification

3.3.1 Latent Variable Models of Network Knowledge

Next, we examined a Multiple Indicator-Multiple Cause (MIMiC) specification to examine if spatial ability may be related to a person's tenure in the metropolitan area or other socio-economic factors. The schematic path diagram for this model is shown in Figure 3-5 below. MIMiC models may be viewed as special cases of the LISREL (Linear Structural Relationships) model, in which the exogenous latent variables of LISREL are observed perfectly.



Source: Ben-Akiva, Ramming and Walker (1999).

Figure 3-5. Relationship Among Traveler Characteristics, Network Knowledge and Indicators.

The structural relationship by which observed variables influence or cause spatial ability, is given by

$\eta = \Gamma X + \zeta$

where

η = network knowledge, a (vector of) latent variables,

Γ = a matrix of structural coefficients,

X = a (vector of) travelers' socio-economic characteristics, and

ζ = a (vector of) random disturbances, one for each element of η.

The measurement of network knowledge is imperfect:

$Y = \Lambda \eta + \varepsilon$

where

Y = a vector of indicators of network knowledge, that is survey responses,

Λ = a matrix of measurement coefficients, and

ε = a (vector of) random disturbances, one for each element of Y.

Once the coefficients Γ and Λ are estimated, fitted values of network knowledge can be used as explanatory variables in route choice models. Ideally, once good model specifications are found, Γ and Λ can be estimated jointly with the utility coefficients using a simultaneous estimation procedure.

3.3.2 Route Choice Given Network Knowledge and Choice Set

Implementation of the Path-Size Logit, MNL, Cross-Nested Logit, C-Logit and IAP Logit have been described in other sections. (PSL is described in section 3.1.2; MNL in section 2.1.3; C-Logit in section 2.1.9; CNL in section 2.1.11; and IAP Logit in section 2.1.13.) Therefore, this section presents an adaptation of the Logit Kernel model to the route choice situation. The Logit Kernel model of Ben-Akiva and Bolduc (1996) combines the Logit and Probit models by adding normal error components to a core MNL model to account for correlation. The specification of error components is based on the topology of paths in the choice set, as will be described below. Recall the general form of the Logit Kernel model (in vector notation) is:

$$\mathbf{U} = \mathbf{X\beta} + \boldsymbol{\varepsilon} = \mathbf{X\beta} + \mathbf{FT\zeta} + \boldsymbol{\nu}$$

where **U** is a J_n by 1 vector of utilities;

$\boldsymbol{\beta}$ is a column vector of K unknown parameters;

X is a J_n by K matrix of explanatory variables;

F is a J_n by M_n factor loading matrix (to be determined);

T is an M_n by M_n lower triangular matrix of unknown parameters (to be determined);

$\boldsymbol{\zeta}$ is an M_n by 1 vector of i.i.d. standard Normal variables; and

$\boldsymbol{\nu}$ is a J_n by 1 vector of i.i.d. Gumbel variables with scale parameter μ.

Therefore,

$$\mathbf{Var}(\boldsymbol{\varepsilon}) = \mathbf{FTT}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}} + (g/\mu^2)\mathbf{I}_{J_n}$$

where g is the variance of a standard Gumbel variable, that is, π^2 / 6.

Elements of **F** and **T** may be estimated or specified from data. One reasonable specification of the Logit Kernel model for route choice would be to assume that the covariance of path utilities are proportional to the length by which paths overlap. This is a common assumption used in implementing the Probit model. We can write

$$\boldsymbol{\Sigma}_n = \mathbf{Var}(\mathbf{FT\zeta}) = \sigma^2 \begin{bmatrix} L_1 & L_{1,2} & \cdots & L_{1,J_n} \\ L_{1,2} & L_2 & \cdots & L_{2,J_n} \\ \vdots & \vdots & \ddots & \vdots \\ L_{1,J_n} & L_{2,J_n} & \cdots & L_{J_n} \end{bmatrix}$$

where σ is a parameter to be estimated.

This specification can be accomplished by setting **F** equal to **Δ**, the link-path incidence matrix. M_n is therefore the number of links in individual n's choice set. Then

$$\mathbf{T} = \sigma \begin{bmatrix} \sqrt{l_1} & 0 & \cdots & 0 \\ 0 & \sqrt{l_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sqrt{l_{M_n}} \end{bmatrix} = \sigma \mathbf{diag}\left(\left[\sqrt{l_1} \quad \sqrt{l_2} \quad \cdots \quad \sqrt{l_{M_n}}\right]^{\mathrm{T}}\right)$$

If the factors $\boldsymbol{\zeta}$ are known, we can write the route choice probability as:

$$P_n(i) = \Lambda(i|\zeta) = \frac{\exp(\mu(X_{in}\beta + F_{in}T\zeta))}{\sum_{j\in C_n}\exp(\mu(X_{jn}\beta + F_{jn}T\zeta))}$$

where X_{in} and F_{in} are the i^{th} row of $\mathbf{X}_n$ and $\mathbf{F}_n$, respectively. Since these factors are not known, the unconditional probability is given by:

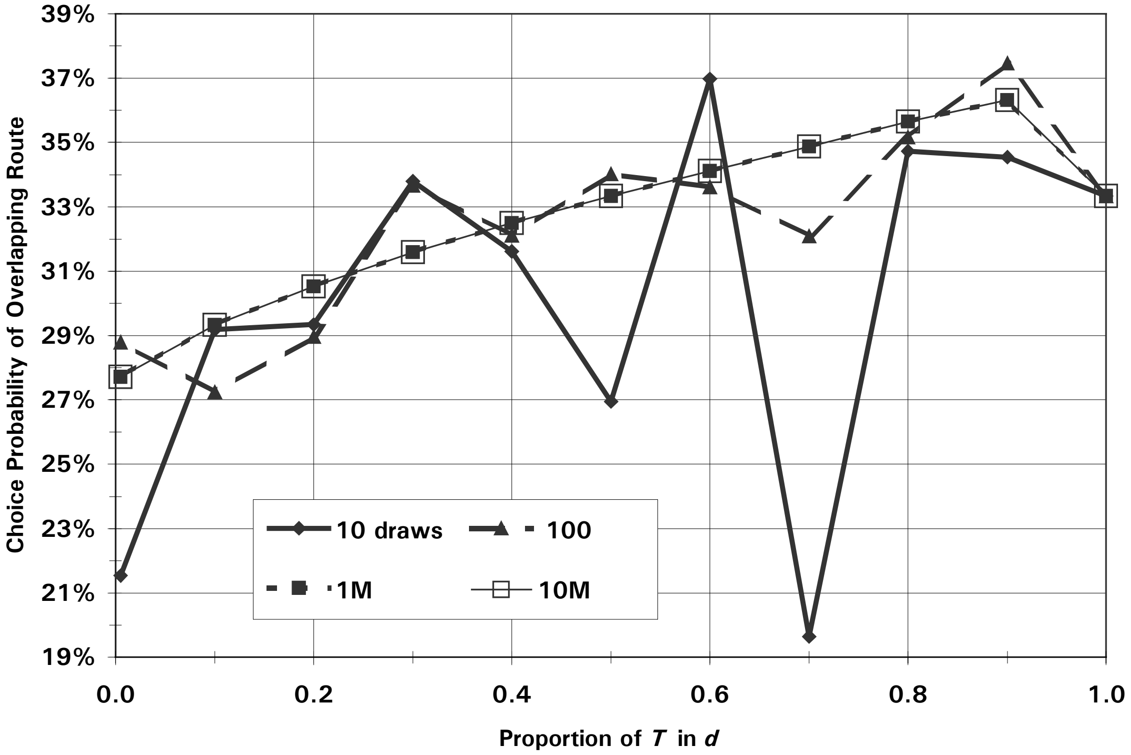
$$P_n(i) = \int_{\zeta} \Lambda(i|\zeta)\prod_{m=1}^{M_n}\phi(\zeta_m)d\zeta_m$$

This probability function can be estimated by simulation:

$$P_n(i) \approx \frac{1}{D}\sum_{d=1}^{D}\Lambda(i|\zeta^d)$$

where D is the number of draws made in the simulation.

The choice of the number of draws has important implications for the quality of the resulting estimates. Figure 3-6 shows a plot of choice probabilities for the overlapping path of the Red-Bus-Blue-Bus network (that is, Figure 2-3) calculated for an LK model with $\sigma = 5$. Probabilities calculated from 10 million draws are used as the basis for comparison. Note that the predicted choice probabilities are quite stable between the 1 million and 10 million draw simulations. Note that the probabilities estimated with only 10 draws have errors of up to 15 percent – about half of the route's choice probability. With 100 draws, the errors are about 3 percent.

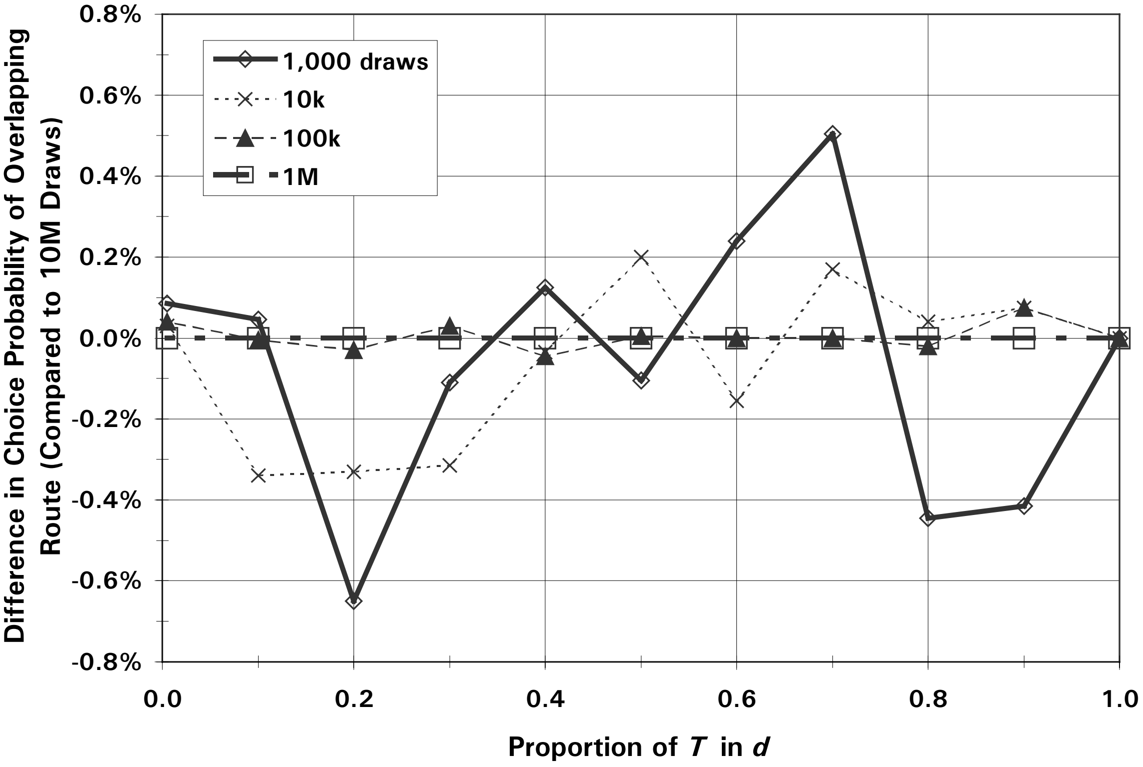


Note: Independent draws are used for each data point.

Figure 3-6. Effect of Number of LK Draws on Choice Probabilities.

To be able to better examine the fit at higher levels of draws, we instead plot the difference between the predicted probability for a given number of draws against our baseline – the probability computed with 10 million draws. This is shown on Figure 3-7. A thousand draws may yield errors of just under a percent for our network. The error reduces to about a half of a percent with 10,000 draws, and about one-tenth a percent with 100,000 draws. This figure also shows that there is little difference between the probabilities calculated with 1 million and 10 million draws.

These results suggest that probit and logit kernel route choice models estimated by simulation should be viewed with some caution. Some empirical studies use as few as 1,000 draws. If this produces about a percentage point error for the Red-Bus-Blue-Bus network – the smallest network in which path overlap is relevant – the error is surely greater in large urban networks.



Note: Independent draws are used for each data point.

Figure 3-7. Comparison of Difference in Choice Probability for Large Numbers of LK Draws.

4
Boston Case Study Data

This chapter describes the various sources of data necessary to estimate the proposed models of spatial knowledge and route choice. A survey of MIT faculty and staff provides the information on travel patterns to be described, and is the topic of section 4.1. This survey also includes questions regarding attitudes toward travel and residence choice, constraints on trip-making, past use of different information sources, and satisfaction with the current means of commuting. Length considerations prohibited self-reporting of alternative routes. Instead, alternative routes needed to be created using engineering time estimates and a consistent set of rules for generating paths. Central Transportation Planning Staff provided a calibrated 1990 auto network, which was used for estimates of typical morning congested travel times. Section 3.2 describes the network model used to develop level-of-service variables for alternative routes.

4.1 1997 Transportation Survey for MIT Employees

The MIT Planning Office periodically conducts campus-wide surveys of travel patterns to collect the necessary data to comply with federal and local commute reduction regulations, assess effectiveness of incentive programs, and plan for future facilities and cost outlays. In November 1997, the Planning Office issued web-based and paper surveys dealing with commuting patterns and attitudes toward travel and using travel information. Three versions of the survey questionnaire were available – one for faculty and staff, one for off-campus students, and one for on-campus students – with questions customized for each population. For example, the faculty and staff survey included questions about inter-city travel and access to Logan Airport. The student versions included greater mention of non-motorized modes, and a section about safety and the SafeRide shuttle.

This research project has focused on the employee survey for three main reasons: (1) The faculty and staff survey produced the greatest number of responses, 1381. (2) This project examines auto route choice behavior and awareness, since automotive trip-making is likely simpler than transit trip-making. That is, transit travel requires knowledge of the location of routes, vehicle headways and schedules, and fare structures. Because faculty and staff are more likely to have greater incomes than students, and are more established in the metropolitan area, they are more likely to own and use autos than are students. (3) Faculty and staff live in more dispersed residences than students, offering more variety in route choice patterns.

4.1.1 Data Available / Questionnaire

The questionnaire for faculty and staff is presented in Appendix B. Initial sections establish home ZIP code, employee classification, office location, availability of parking and transit pass, weekday and weekend trip frequency, and the need and reason for mid-day trips. A separate section examines the multi-modal description of the route the respondent uses most frequently, the respondents' satisfaction with various aspects of the journey, how the respondent learned that route, and the frequency with which the respondent uses other routes. Other sections elicit attitudinal responses to seeking information about travel conditions, preferences about various modes, constraints to travel and motivations for selecting their residence. The final section of the survey requests demographic information such as gender, income, race, and tenure in the metropolitan area, at MIT and at the respondent's current residence.

4.1.2 Cleaning

Raw survey data were received from the Planning Office in ASCII format and converted to a SAS data set. Responses to likert attitudinal questions and home ZIP code were checked for reasonableness; in some cases, responses were replaced with missing value indicators. For some observations, transposed numbers, extra digits or missing digits of ZIP codes were corrected. Records that appeared to be duplicates were deleted. (In some cases, this may have been caused by the respondent pressing the "submit" button midway through completing the survey, thinking it would save their responses, then using their web browser's "back" button to return and complete the survey. In these cases, only the last observation was retained.)

Some ZIP codes from beyond New England were reported in the survey. These responses were believed to be researchers and professors with joint appointments who maintain homes near their other affiliated institute. Such people may have reported "commuting" by airplane, but presumably these trips are made infrequently. Because these people were not believed to be regular commuters, they were excluded from further analysis.

Next, possible means of learning about the primary commuting route were examined. Respondents could indicate any combination of eight means of learning their route:
(1) consulting a map, (2) consulting transit schedules, (3) consulting a telephone information line, (4) asking friends, relatives or colleagues, (5) attempting to navigate when needed, referring to signs and strangers' advice as necessary, (6) exploring the city during spare time, (7) learning about the city through experience prior to working at MIT, and (8) a response indicating the traveler could not remember how they learned the route. Because a "can't remember" option was available, respondents who did not indicate at least one means of learning were flagged and eliminated from further consideration.

We also dropped respondents who reported needing to care for a relative (child or older adult), as the location of day care facilities was believed to constrain these people's route choices. Specifically, individuals who indicated that the need to care for a relative affected their route choice "not at all" or "very little" were included. Those reporting that this responsibility had "little," "much" or "very much" impact on their route choice were excluded.

Next, mode segments were examined to classify each respondents' travel patterns. Respondents were asked to report the sequence in which they used various modes in describing their primary commuting route to MIT. These responses were converted to a label for easier editing (such as auto-to-subway-to-bus), and grouped into larger categories: auto only, transit only, auto and transit, or other (primarily walking and bicycling). Inspection of the resulting labels revealed some illogical mode sequences, such as subway-to-auto-to-bicycle. Some respondents appeared to misinterpret the question about mode sequences and instead provided a ranking of single-mode trips in decreasing frequency of use. For example, a person who uses the subway most often, followed by driving alone, and bicycling only occasionally, may have produced the subway-to-auto-to-bicycle label. Such responses were recoded. Mode sequences were also checked against reported walking, waiting, riding and total travel times, and the text description of the automotive portion of the route. For example, employees claiming to use transit were expected to report at least some waiting time.

Respondents were allowed to provide open-ended responses to the questions about tenure in the metropolitan area, at MIT and at their current residence; that is, the web survey forms could not restrict responses to only numeric values. Further, because of a web coding error, the responses to the number of years worked at MIT and the number of years lived at the current residence were concatenated. These three variables were manually examined to separate the years worked at MIT and years lived at current residence variables. In some cases, respondents volunteered extra information, such as the number of months, or a modifier to years such as "+", "–", "<" or ">". When months were given, this information was converted to a decimal number of years. Modifiers resulted in a half year being added or subtracted from the number of years reported. Also, the number of years in the metro area was required to be at least the number of years at MIT or at the current residence, because the "metro area" was defined as the cities and towns with economic links to the Boston center city, and thus includes the residences of all current MIT employees.

We are particularly interested in respondents' tenure in the metropolitan area, because we believe this provides a useful explanatory variable for estimating network knowledge. We hypothesize that a new resident to a metropolitan area – even one possessing map ability – will have little network knowledge, as she or he has not had an opportunity to learn the area. That is, she or he may initially acquire only landmark knowledge during the first few days in the city. Initially, he or she may get routes from others – friends, relatives, colleagues, taxi drivers or information lines. Learning where the landmarks are will be required to follow the route instructions. As the resident has more exposure to the urban area, the traveler may learn a few routes. This traveler may also look at maps, which may speed integration of data about the new city into map knowledge, if he or she possesses map ability. After some time, the traveler may integrate the route experiences and map information into a mental map. Stern and Leiser (1988) present some evidence that a driver may stop acquiring network knowledge after a period of years, when the driver believes he or she has sufficient knowledge for navigating.

Descriptions of auto routes were examined to be coded within the computer network representation described in section 4.2 below. A single point (centroid) was originally chosen to represent the destination of all respondents on campus; this point corresponds to the main campus cluster of buildings around the Infinite Corridor. Although office location is available from the employee survey, most faculty and staff are not able to park adjacent to their building. Information about which parking facility an individual was assigned to was not requested in the survey; however, a few respondents volunteered this information in the description of their commuting route. To improve the ability of the path generation algorithms to duplicate the observed paths, we coded multiple centroids representing MIT parking facilities and turning movements. (That is, turn penalties were applied so that each destination centroid could only be reached from one approach link. Therefore, multiple centroids represent the same parking facility. Turn penalties are discussed further in section 4.2.3.) Respondents were assigned to destination centroids based on the last street described in their route. If that street was within the MIT campus, a destination was selected so that the route was constrained to use the last reported street. If the last reported street was off campus, the least travel time path was constructed from that street to the center of campus (Main Lot). If the least-time path went past any other parking facility first, that parking facility was used as the respondent’s destination.

Likewise, all respondents from the same ZIP code were initially assigned to a single centroid. However, in several cases, it was possible to infer a subarea within the ZIP code where the respondent began his or her journey from the first street or road listed. During later revisions to improve the performance of the choice set generation algorithms, respondents were assigned to various centroids within a ZIP code that allowed the most direct access to the first segment in their route.

To the extent possible, routes followed the facilities listed in the route description; when the transition from one facility to another was ambiguous, an attempt was made to reconstruct the route minimizing the distance off listed streets, perhaps following paths of other respondents traveling in the same corridor. In remaining ambiguous cases, least congested time paths were used, to improve the ability of the path generation algorithm described below (which also minimizes congested time) to match survey paths. If respondents provided the evening route, that route was reversed to produce the morning route. When multiple routes were listed, the first route listed was assumed to be the primary route. In some cases, respondents reported using facilities not included in the computer network representation; such people were coded as using the nearest parallel facility. For example, several people reported using Harvard St. in Cambridge, which isn't in the network; these people were coded as using Broadway in Cambridge. If a route could not be confidently constructed from the information given in the open-ended survey question, that respondent was dropped from further consideration.

4.1.3 Summary Statistics

From the 1,381 total responses to the faculty and staff survey, records were screened to establish their eligibility for this modeling exercise. Respondents needed to have a ZIP code in New England, travel by auto all the way to MIT as their primary commuting means, have no travel constraints such as child care or care for a family member to stop off at during the commute, and to have provided usable responses about commute characteristics and the attitudinal questions described above. One-hundred eighty-eight respondents met these screening criteria and thus formed the origin-destination pairs on which the various route generation algorithms described in Section 4.2 were performed.

Some frequency counts of descriptive socioeconomic variables are shown in Table 4-1 for the 188 respondents used in route generation modeling. These respondents live in a variety of areas throughout the Boston metropolitan area. Only about seven percent live in Cambridge, and an additional 31 percent live in the cities and towns that share a border with Cambridge – Boston, Arlington, Belmont, Brookline, Somerville and Watertown. Therefore over 60 percent of the 188

Response	Number	Percent
Location of Residence		
Cambridge	13	7%
Abutting Communities (See Note on Page 89)	58	31%
Other Communities	117	62%
Housing Tenure		
Own	128	68%
Rent or Lease	57	30%
No Response	3	2%
Gender		
Female	110	59%
Male	78	41%
Annual Household Income, Before Taxes		
Less than $25,000	1	1%
$25,000 to $49,999	47	25%
$50,000 to $74,999	49	26%
$75,000 to $99,999	28	15%
$100,000 to $149,999	33	18%
$150,000 or More	13	7%
No Response	17	9%
Age		
21 to 24 Years	6	3%
25 to 29 Years	13	7%
30 to 39 Years	59	31%
40 to 49 Years	56	30%
50 Years or Older	53	28%
No Response	1	1%
Marital Status		
Single, Separated, Divorced or Widowed	69	37%
Married or Long-Term Relationship	118	63%
No Response	1	1%

Table 4-1. Selected Characteristics of Survey Respondents.

Response	Number	Percent
Number of Children Living at Home		
None	131	70%
One	25	13%
Two	21	11%
Three or Four	5	3%
No Response	6	3%
MIT Employee Classification		
Faculty or Medical Staff	20	11%
Administrative Staff	94	50%
Other Academic Staff	6	3%
Support Staff	37	20%
Research Staff	15	8%
Service Staff	5	3%
Other Classifications or No Response	11	6%
Type of Appointment		
Full-Time	169	90%
Part-Time	11	6%
Visiting or Retired	4	2%
No Response	4	2%
Commuting Occupancy		
Drive Alone	139	74%
High-Occupancy Vehicle (2 or More Occupants)	49	26%

Table 4-1. Selected Characteristics of Survey Respondents (Continued).

Response	Number	Percent
Day-to-Day Variation of Arrival Time		
No Variation	4	2%
No More Than 15 Minutes	53	28%
No More Than 30 Minutes	66	35%
No More Than an Hour	21	11%
More Than an Hour, But Only in Unusual Circumstances	14	7%
Often More Than an Hour	17	9%
Depends on Class or Meeting Schedule	11	6%
No Response	2	1%
Availability of a More Flexible Schedule		
Yes	112	60%
No	50	27%
Not Sure or No Response	26	14%

Notes: Based on 188 responses from auto travelers with usable information about their commute, who did not day care stops or other constraints on route choice, and who responded to the questions about learning their route and seven key attitudinal questions.

Total percentages may not add to 100 percent because of rounding.

Abutting Communities are Arlington, Belmont, Boston, Brookline, Somerville and Watertown.

Source: MIT Planning Office (1997)

Table 4-1. Selected Characteristics of Survey Respondents (Continued).

respondents live in more distant communities. Of the 188 respondents, 19 live in Boston, the most of any single community. Cambridge and Arlington are the residences of 13 respondents each, followed by Watertown and Newton with 11 respondents each. Over two-thirds own their own home, or are paying off a mortgage used to purchase the home. Sixty percent of the sample are women.

Because many respondents were resistant to providing their race, and there was not much diversity among those who did respond, race was not used for route choice modeling.

A broad spectrum of household incomes are represented; over 80 percent of the survey respondents have household incomes (before taxes) within the range of $25,000 per year to $150,000 per year. Almost half the respondents have annual household incomes of $75,000 or more. The effect of survey length on participation was a concern, as those with higher values of time – likely corresponding to higher incomes – may have been discouraged from completing the questionnaire. However, the broad income distribution suggests this is not the case. Alternatively, low income people may be equally pressed for time, because low hourly wage rates result in incentives to work longer hours, leaving less time for other activities.

About ninety percent of the respondents are 30 years of age or older, with roughly equal proportions being in their thirties, in their forties, or aged fifty or older. Older individuals are of course more likely to have lived in Boston for a longer time and thus know more of the street network. Also, such individuals may be less likely to try receiving traffic information from new sources such as the SmarTraveler telephone system. The distribution of the number of years lived in the Boston metropolitan area is shown in Figure 4-1. The median tenure is just under 20 years. About one-fifth of the respondents have lived in the area five years or less. Another tenth have lived here 40 to 65 (the maximum reported) years.

About two-thirds of the respondents have partners, and over a quarter have one or more children living at home. People living in larger households may be aware of more routes, since they might learn new routes from other household members. However, single people may have roommates to share housing expenses, and could learn new routes from roommates in much the same way that partnered individuals may learn new routes from family members. The 1997 survey does not allow identification of people who live with non-relatives or extended family members.

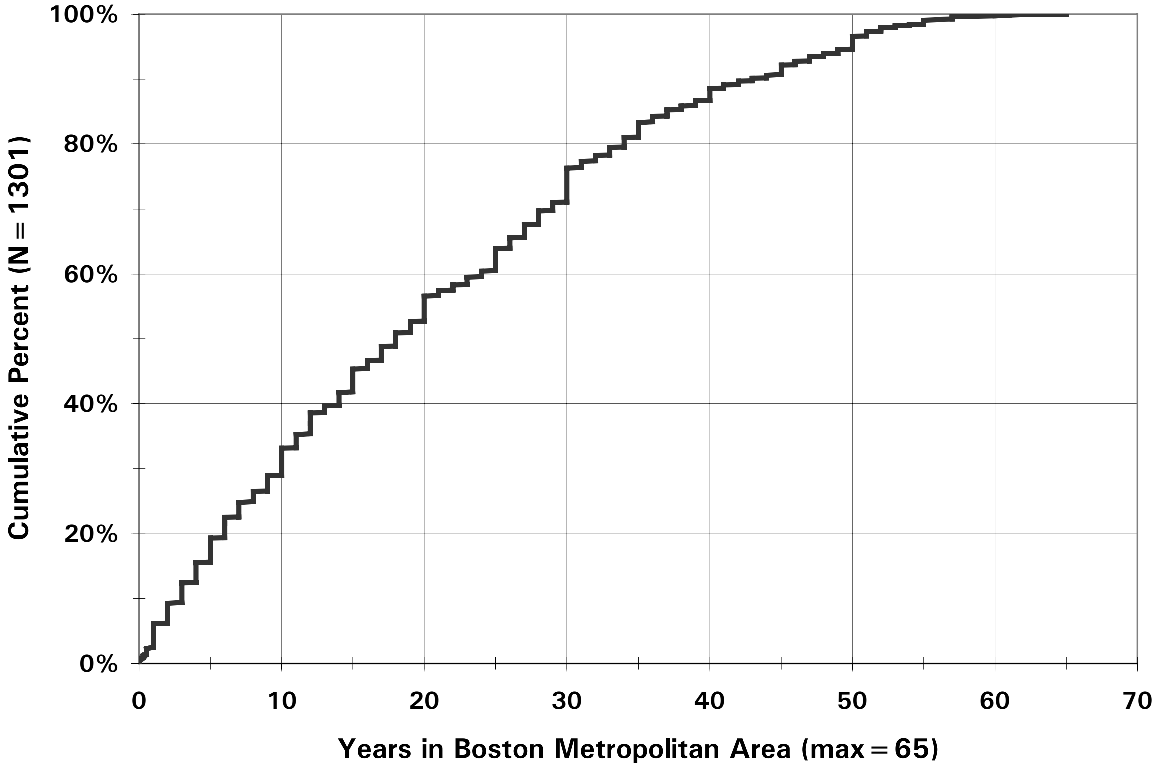


Figure 4-1. Distribution of Tenure in Boston Metropolitan Area.

Half of the survey respondents are administrative staff. This seems a large overrepresentation, since administrators make up about 17 percent of the MIT workforce. (Table 4-2 lists numbers and percentages of MIT employees who work at the Cambridge campus by classification.) However, "other academic staff" seem to be underrepresented. It is possible that many members of this official category chose the administrative classification, which has a more intuitive or personal connotation. These two categories make up 53 percent in survey, versus 47 percent in population, which is a more reasonable correspondence. About a fifth of the 188 respondents are support staff members, which compares well with the proportion of the MIT workforce. A web coding error prevented faculty and medical staff members from being distinguished with confidence. Together, faculty and medical staff form 11 percent of the route choice sample, compared to 14 percent of the MIT workforce. Research staff are 8 percent of the sample, and 11 percent of the MIT workforce. Both of these percentages are within the expected survey margin of error. Service staff form 3 percent of the sample, but a tenth of the workforce. Several factors may contribute to the under-representation of service staff: they may have limited internet access, and were unwilling to request paper surveys. Service staff workers may also have lower incomes than say faculty or administrators, and thus be more likely to take transit. Such service staff members would not appear in the auto route choice sample.

Classification	Number			Percent		
	Men	Women	All	Men	Women	All
Administrators and Senior Officers	558	752	1,310	12%	25%	17%
Faculty	775	135	910	16%	4%	12%
Medical Staff	73	75	148	2%	2%	2%
Other Academic Staff	1,796	520	2,316	38%	17%	30%
Researchers	592	302	894	13%	10%	12%
Service Staff	618	136	754	13%	4%	10%
Support Staff	291	1,140	1,431	6%	37%	18%
Total	**4,703**	**3,060**	**7,763**	**100%**	**100%**	**100%**

Note: Percentages do not add to 100 percent because of rounding.
Source: MIT Personnel Office (1997)

Table 4-2. Employee Counts at MIT Cambridge Campus, October 1997.

The vast majority (90 percent) of respondents work full time. Another six percent work part time. However, most of the part-time workers at MIT are scheduled to be on campus four to six hours all five days of a week, rather than for two or three long workdays each week. Therefore, we would not expect part-time employees to gain network knowledge more slowly than full-time employees.

Almost three-quarters of the auto route choice sample commute by single-occupancy vehicle (SOV). This is consistent with other MIT Planning Office statistics that suggest about 36 percent of staff members arrive by SOV while about half use modes other than private autos. We might expect high-occupancy vehicle (HOV) commuters to gain network knowledge by sharing experiences in much the way that members of large households would.

Some attitudinal responses are presented in Figure 4-2. Almost 90 percent of respondents tend to use one route for most of their commuting. About 70 percent are satisfied with their commuting route, though roughly the same proportion dislikes city driving. About 80 percent reported they were knowledgeable enough to find an alternative route.

Respondents were asked to consider how they became aware of the primary route they use for commuting to MIT. (See Table 4-3.) The largest majority, 58 percent, said their experience from other jobs, residences, events, etc. in the region allowed them to become aware of their current habitual route. About a third of the 188 respondents reported learning their current route from a map. Maps were the second most common source of learning, after experience. Asking friends, relatives or colleagues was the third most common response, which just over thirty percent cited.

Just under a fifth of the respondents reported they learned their route by exploring Cambridge and the surrounding area during their free time. In contrast, only 5 percent say they learned their route by exploration when it was necessary to undertake the trip. These results are consistent with individuals wishing to minimize the chance of being late by not traveling in unfamiliar areas while under time pressure. A few respondents said they learned their route from transit schedules or a travel information line, which was unexpected. It is unclear whether these respondents misunderstood the question, or perhaps are former bus riders who adapted the same route when they switched to using a private automobile. We do not believe the presence of these unexpected responses adversely effects the validity of the more relevant means of learning, such as maps.

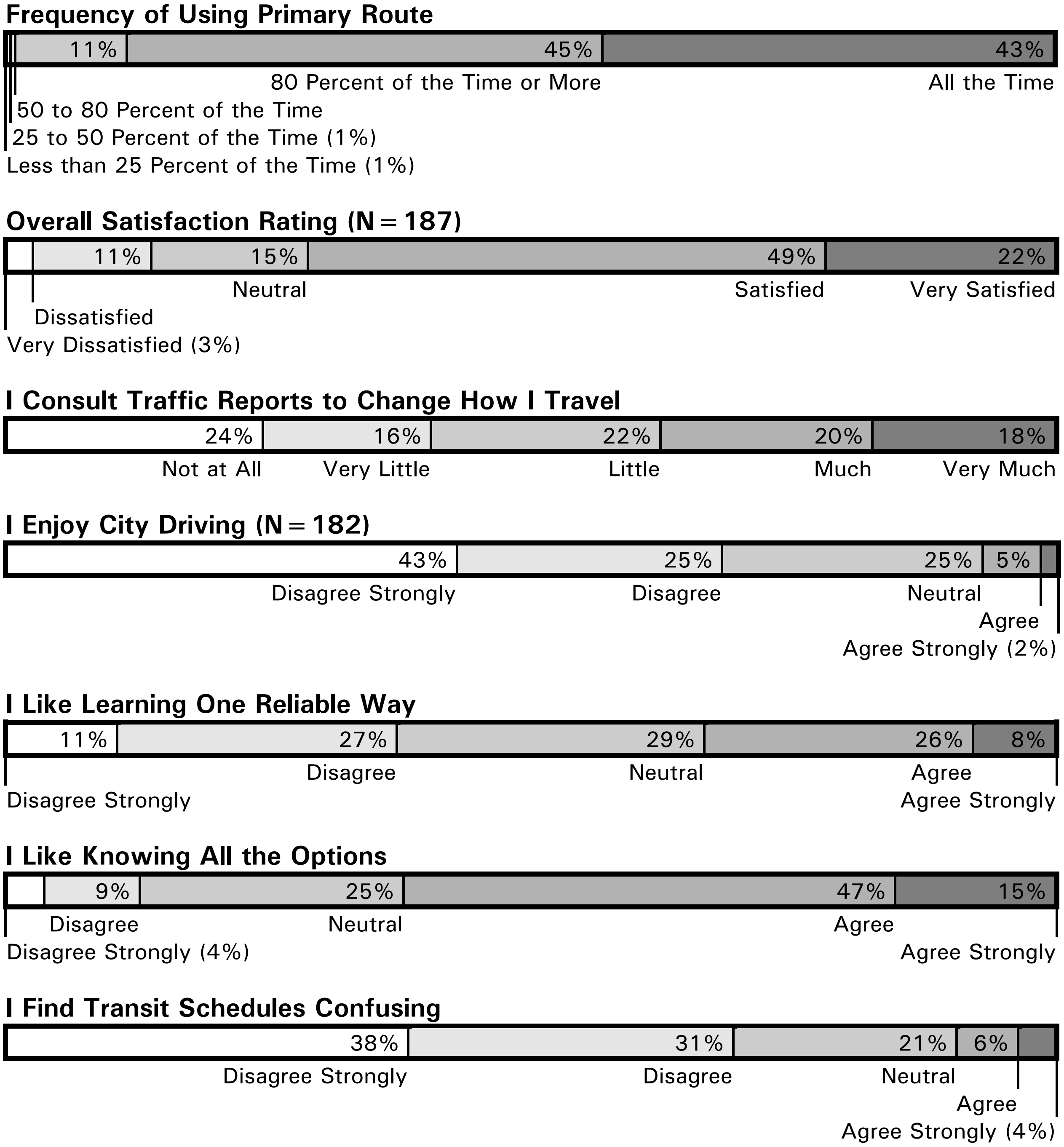
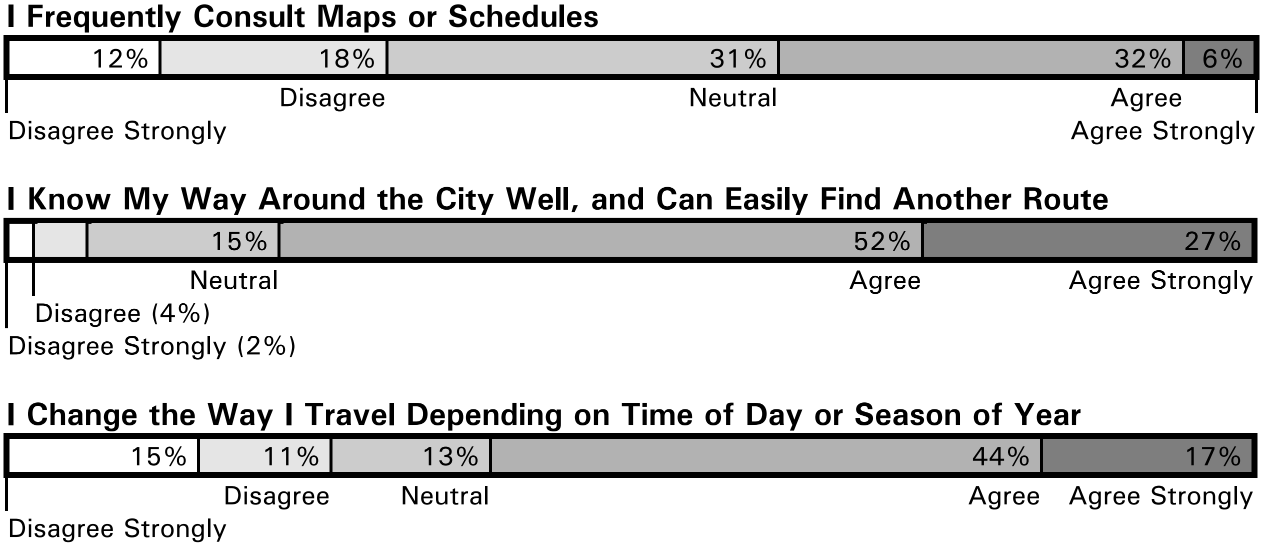


Figure 4-2. Summary of Commuting Behavior and Attitudes.



Notes: Percentages do not add to 100 percent because of rounding. N = 188 unless otherwise specified.

Source: MIT Planning Office (1997)

Figure 4-2. Summary of Commuting Behavior and Attitudes (Continued).

4.2 Network Data

Estimating a route choice model requires information about the various attributes of both chosen and alternative routes. The design of the 1997 Transportation Survey did not allow for collection of data regarding alternative routes that respondents might consider. Instead, we adopted an engineering approach to constructing possible alternative routes for all respondents. That is, the computer network that Central Transportation Planning Staff (CTPS), the Boston MPO, uses for planning purposes was requested to provide estimates of travel times on various roadways throughout the region. Section 4.2.1 describes the structure of this network and the variables supplied by CTPS. The following sections describe the variables we added to this network (section 4.2.2) and turn penalty refinements that were made to produce more reasonable paths (section 4.2.3).

Means of Learning	Selected Auto Commuters		All Faculty and Staff	
	Number	Percent	Number	Percent
Experience from other jobs, residences, attending various events, etc.	109	58%	591	43%
Using a map	62	33%	300	22%
Asking friends, relatives or colleagues	58	31%	287	21%
Exploring the city during spare time	33	18%	189	14%
Trying to find the way by following signs or asking strangers when necessary	10	5%	37	3%
Using transit schedules	2	1%	165	12%
Calling an information line	2	1%	33	2%
Can't remember	10	5%	84	6%
Population Totals	188	100%	1381	100%

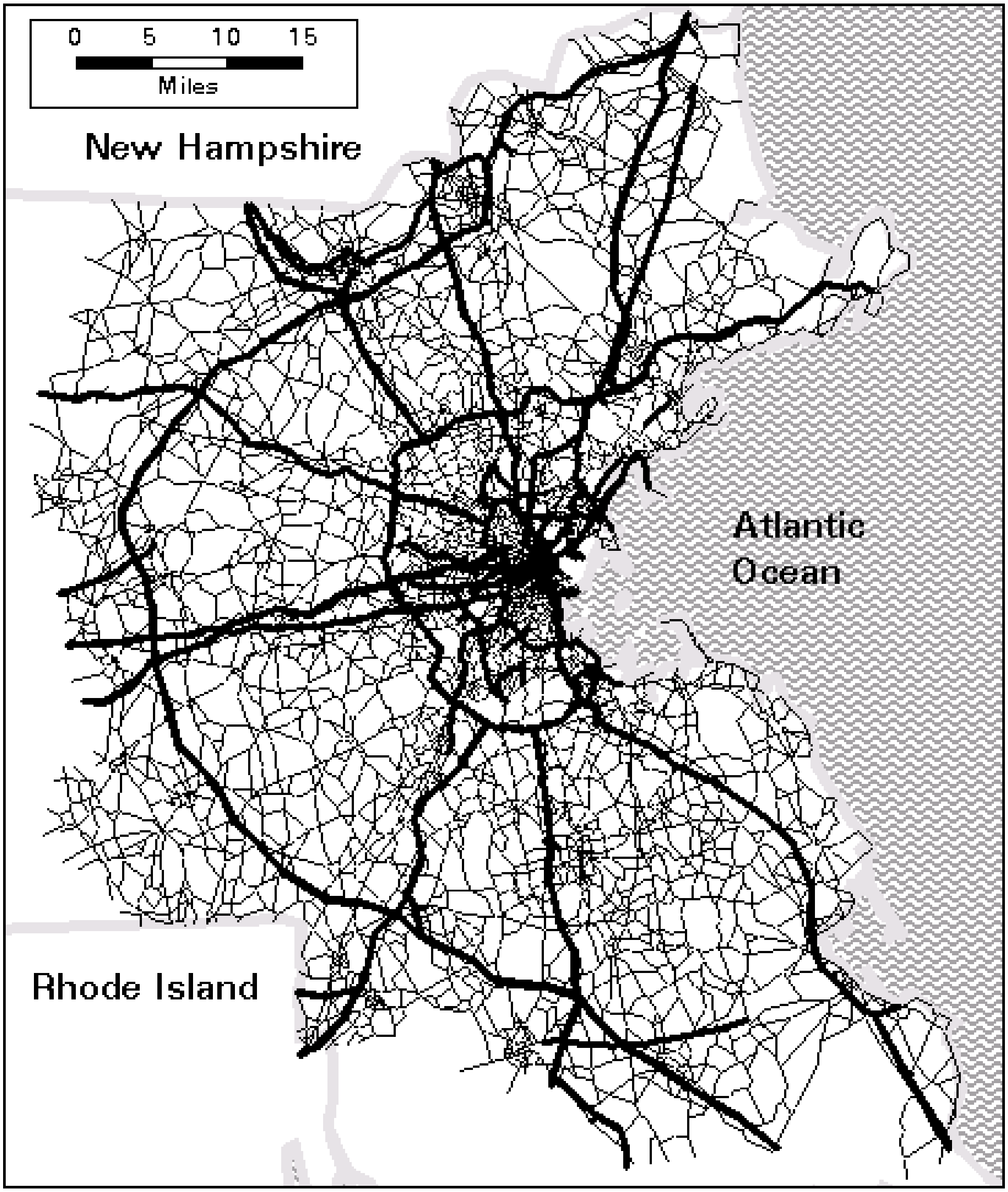
Notes: Multiple responses were allowed to the question "How did you find out about this way of coming to MIT?" Some respondents did not indicate any of the eight reasons; such respondents were not included in the auto commuter sample for route choice modeling.

Source: MIT Planning Office (1997)

Table 4-3. Means by Which Respondents Learned Their Primary Commuting Route.

4.2.1 CTPS 1990 Highway Network

The CTPS highway network calibrated for 1990 is used to estimate auto travel times under congested conditions and to identify possible alternative routes. It contains 787 internal zone centroids, 101 external cordon crossing stations, 12,089 other nodes and 19,148 links. The network corresponds to a study area bounded by the New Hampshire border on the north, a line just inside Interstates 190 and 395 on the west, and the Rhode Island border and a line south of US 44 on the south. All physical links in the network are shown in Figure 4-3. Virtually all



Note: Freeways and expressways are shown in thicker lines.
Source: Peterson (1998)

Figure 4-3. CTPS 1990 Highway Network for Eastern Massachusetts.

individual streets are represented in Downtown Boston and the Back Bay. However, in the outer suburbs, only major roads are represented in the network, at a density of roughly one to five miles between intersections. Limited-access freeways are composed of two sets of one-way links, and ramps are explicitly coded.

CTPS used the mainframe UTPS program suite to conduct a 1990 traffic assignment for calibration. The 1990 assignment, based on trip generation from census population figures for that year, was the most recent available. Network link and node information was transmitted to MIT as a downloaded ASCII file. We then imported this file to the PC-based TransCAD system to allow for visual editing of the network and coding of various route choice algorithms.

As part of the conversion, node coordinates needed to be converted to longitudes and latitudes, as required for TransCAD. The UTPS node coordinates were based on northward and eastward departures from the 1927 North American Datum for the Massachusetts Mainland, divided by 100 feet. TransCAD includes options for multiplying input coordinates by 100 (or any other factor) and applying the resulting coordinates to various state surveying systems to arrive at longitude and latitude values.

Network link variables that were included in the text file transmitted by CTPS are described in Table 4-4. Note that we expected many of these variables to be related to the prominence of a route. For example, roads with higher capacities (on a per-lane basis) are more likely freeways, or high performance roadways, and therefore, we expect less familiar drivers to be aware of higher capacity roadways. Similarly, roads with more lanes would be more likely to be highlighted on maps, and therefore less knowledgeable drivers would be more likely to know them.

The CTPS network also contains several variables related to impedance, such as distance, free-flow time and estimated time. Clearly, these variables affect route utility. Since we expect travelers to prefer shorter (quicker) paths – and therefore to seek out information about such

Name	Range	Description
ANODE	1-16033	ID of node at “tail” of link
BNODE	1-16033	ID of node at “head” of link
CAPPERLANE	300-2700, 9999*	Capacity of link in vehicles per hour per lane
LANES	1-4, 7*	Number of lanes in one direction
DISTANCE	0-6.8	Distance in miles
FF_TIME	0-7.53; 0-65*	Free flow time in minutes
C_SPEED	Undefined, 0.86-60	Congested (a.m. peak hour) speed in mph
C_TIME	0-7.55; 0-65*	Congested (a.m. peak hour) travel time in minutes
TOWN_CODE	Blank, 1-350, 776-888*	Code for the town in which a link is located
DIR_CODE	1 or 2	Directionality code, 1 = one-way ANODE to BNODE, 2 = two-way

Note: * indicates values used only on centroid connector and other “virtual” links
Source: Peterson (1998)

Table 4-4. Variables Supplied in the CTPS Network.

paths – we expect these variables to be useful in Path Size and other formulations. Note that theoretical reasons suggest against using a Path Size formulation based on estimated time – Path Size should represent drivers’ perceptions of the network, and should therefore be independent of demand and congestion levels.

4.2.2 Additional Link Attributes

We coded many additional variables that we suspected might have an impact on travelers’ awareness or choice of routes. First, centroid connector links were identified by those links having an ANODE or BNODE of 888 or lower. Centroid connector links do not necessarily correspond to any local street, but are rather an abstract representation of the local street grid. They allow travelers from diverse origins to be loaded onto the computer network representation, which contains more major streets. Therefore, centroid connectors are given special treatment in the path generation and route choice modeling process.

We are interested in the functional classification of various links because we expect inexperienced travelers to be more likely to know routes with "higher" classifications, such as freeways and expressways. We identified freeways by selecting links with capacities of 1,650 vehicles per lane per hour or greater, excluding centroid connectors. The set of freeways was then visually inspected to insure continuity and that all Interstate routes were included.

After coding freeways, expressways were identified based on the following criteria:

- routes having interchanges but not full grade separation and access control (thus failing to qualify as freeways),
- parkways maintained by the Metropolitan District Commission, and
- roadways given special treatment on highway maps or atlases.

Next, we attempted to identify neighborhood factors that might affect route choice. From Anchor Point theory, we expect that drivers know more roadways in their town of residence and where they work. In other towns, we would expect their mental maps to be limited to the more prominent roads. Also, perceived neighborhood security might also be correlated with prominence. We use the term "security" to refer to avoiding risk of crime, while "safety" refers to minimizing risk of a crash. Unfamiliar travelers will not seek out routes in communities they consider to be insecure.

The City of Cambridge publishes comprehensive crime statistics for each of its 13 neighborhoods. Crimes are classified by the city as housebreaks, street robberies, auto thefts, larcenies from motor vehicles, malicious destruction of property, and drug arrests. Because crime data at this level of detail was not as available for other towns in Eastern Massachusetts, we chose to develop an instrumental variable from readily-available census data. We believe that auto thefts, larcenies from motor vehicles, malicious destruction of property and street robberies would be most correlated with drivers' reluctance to use particular streets. Visual inspection suggested that these crimes correlated well with the unemployment rate.

An unemployment rate of six percent (calculated excluding those not in the civilian workforce) appeared to correspond to the areas of Cambridge popularly believed to be the least safe. Therefore, the six percent unemployment cutoff was applied to ZIP codes outside Cambridge. The affected ZIP codes included downtown Boston, Dorchester Center, East Boston, Grove Hall, Mattapan, Mission Hill, Roxbury, the South End, Uphams Corner, Boxborough, Blackstone, Brockton's east side, Chelsea, Lawrence, Lowell, Central Square and southeastern Lynn, Taunton and Tyngsborough. Non-freeway links entirely within such ZIP codes were coded as "insecure." We chose to exempt freeways from the security designation, because freeways are limited access facilities and fenced off from pedestrians. Travelers on freeways therefore have little interaction with the surrounding neighborhood.

Because the network data supplied by CTPS contained congested travel speeds and times, but not assigned volumes, we estimated volumes by using the inverse of the Bureau of Public Roads (BPR) function. That is,

$$V = C\left[\frac{1}{\alpha}\left(\frac{T_c}{T_0} - 1\right)\right]^{1/\beta}.$$

where V is the hourly volume,
C is the hourly "practical" capacity,
T_c is the congested travel time of a link,
T_0 is the free-flow travel time, and
α and β are parameters.

Since CTPS was unable to locate documentation for the assignment procedure, we assumed default parameter values ($\alpha = 0.15$ and $\beta = 4$).

The toll facilities in Eastern Massachusetts are the Massachusetts Turnpike (I-90), the Tobin Bridge over the Mystic River (U.S. 1), and the Sumner Tunnel (Route 1A). Tolls for these facilities were coded based on information from publicly-available sources. Outside Route 128 (I-95), the Massachusetts Turnpike uses a ticket system with distance-based tolls. (See Massachusetts Turnpike, 1998.) Passenger vehicle tolls were converted to link-based tolls by examining the difference in cost for Boston-bound trips that enter at various MassPike interchanges. Barrier tolls were coded on the links where they occur. Tolls are collected on the Tobin Bridge and Sumner Tunnel in the inbound (that is, south-west to Downtown Boston) direction only. Passenger vehicle tolls had been at $0.50 and $1.00, respectively, before their tolls doubled when the Ted Williams Tunnel opened in 1998. Tolls from 1997 were used to insure consistency with the MIT Transportation Survey. Since survey respondents may choose to go home by routes other than the reverse of the a.m. commute, tolls were coded with the one-way trip price only (as opposed to adjusting for half or full daily round-trip cost).

From a utility perspective, travelers will avoid tolls to economize. However, there are very few toll roads in Eastern Massachusetts, and they tend to be high-capacity roadways such as the Mass. Pike, Tobin Bridge, and Sumner and Callahan Tunnels, which are more prominent.

Names and numbered routes were coded by hand, while referring to Universal Publishing Co., Inc. (1997) and Rand McNally (1994). Numbered routes were identified in a binary variable, and the interaction (product) of that variable with congested travel time was calculated. Since many survey respondents used route numbers – or the familiar names (e.g., Mass. Pike or Southeast Expressway) of government-designated highways – to identify the facilities in their commute path, we theorized that presence of a state or federal route number would be correlated with increased information for following that route. Therefore, travelers would perceive such facilities as more prominent.

The additional variables are summarized in Table 4-5.

Name	Range	Description
CAPACITY	300-8000, 999999*	Capacity of all lanes in one direction, vehicles per hour; CAPPERLANE*LANES
CENT_CONN	0, 1* or 3*	Centroid connector code; 0 = "real" link, 1 = centroid connector, 3 = centroid connector not used in path-building process
ONE	1	The constant one
HIERARCHY_	1-4, 8, 9	Hierarchy number; 1 = freeway or ramp, 2 = expressway or ramp, 3 = major arterial, 4 = local road, 8 = centroid connector, 9 = not used in network
FREEWAY	0 or 1	Freeway dummy variable; 1 = freeway or ramp, 0 otherwise
FWY_TIME	0-6.8	FREEWAY*C_TIME
EXPWY	0 or 1	Expressway dummy variable; 1 = expressway including MDC parkway, 0 otherwise
EXPWY_TIME		EXPWY*C_TIME
UNSAFE	0 or 1	Instrumented insecure link dummy variable; 1 = unemployment rate is greater than 6 percent in ZIP code or Cambridge Neighborhood, 0 otherwise
UNSF_TIME		UNSAFE*C_TIME
NUM_RT	0 or 1	Numbered route dummy variable; 1 = link is part of a state numbered route (e.g., Route 2, US 1, I-95), 0 otherwise
NUMRT_TIME		NUM_RT*C_TIME
NAME	A 10-character string	Facility name
TOLL	0-1	One-way toll in dollars
IMP_VC	0-	Volume to capacity ratio implied by BPR relation
IMP_VOL		Volume implied by BPR relation
REAL_RD	0 or 1	"Real" road dummy; 1 = "real" road (CENT_CONN=0), 0 otherwise

Table 4-5. Variables Added to the Network.

Name	Range	Description
USED_CC	0 or 1	Used centroid connector dummy; 1 = centroid connector is used by path-building routine (CENT_CONN=1), 0 otherwise
MIT_CC	0 or 1	MIT centroid connector dummy used by path-building routine; 1 = link is a centroid connector to the TAZ representing MIT, 0 otherwise
ORIG_CC	0 or 1	Origin centroid connector dummy used by path-building routine; 1 = link is a centroid connector to the current origin, 0 otherwise
DEST_CC	0 or 1	Destination centroid connector dummy used by path-building routine; 1 = link is a centroid connector to the current destination, 0 otherwise
X_WATER	0 or 1	Water crossing dummy: 1 if link represents a bridge or tunnel, 0 otherwise

Table 4-5. Variables Added to the Network. (continued)

4.2.3 Turn Penalties

In addition to creating extra link attributes, we also coded a turn penalty file. (CTPS reported that the network we were given did have a turn penalty file, but they were unable to extract it from their archives.) Generally, these turn penalties corresponded to movements that are illegal or physically impossible to make. In some cases, the schematic representation of clover-leaf intersections required turn prohibitions to reflect the actual configuration of ramps. Also, turn prohibitions were applied to prevent U turns at the ends of divided highway segments and at exits to rotaries.

The implementation of turn penalties in TransCAD required further treatment. TransCAD allows one of three regimes for turn penalties to be used Caliper Corporation (1996b):

1. *Generic turn penalties*. Penalties are based solely on the angle of incidence between links being traversed. That is, a single time penalty applies for all left turns in the network. Similarly, penalty values can be specified for all right turns, U turns and through movements.

2. *Link-type turn penalties*. In this regime, turn penalties are based on the direction of movement between two links, and the value of a specified attribute of the links. This link type variable typically corresponds to code designating the functional type of a link (e.g., ramps, freeways or arterials in a highway network; or access links, bus route segments and rail lines in a transit network).

3. *Intersection-specific turn penalties*. In this regime, the turn penalty file has one record for each link-to-link movement.

We desired a hybrid approach, where certain movements would be prohibited, but remaining generic movements (e.g., left turns and U turns) could be penalized to produce more reasonable paths. However, TransCAD does not allow turn penalties to be additive. That is, if any intersection-specific penalties are used, the settings for generic and link-type penalties are ignored. Therefore, we developed a program to explicitly list all turn movements between all physical links (that is, excluding centroid connectors) in the network. The program also calculated the angle of turn between the two links. For example, a zero-degree angle indicates continuing straight, while a U turn is a (+/-) 180 degree turn. Negative numbers indicate left turns. Turns could then be grouped based on direction and severity of the turn. The turn penalties used in the identification of shortest paths are shown in Table 4-6. Freeway and expressway links were excluded from generic, directional turn penalties. In some cases, multiple freeway or expressway links were coded between interchanges, to better represent roadway geometrics. The nodes between such links would not be perceived by drivers as turns, because the only option at that node is to continue on the current facility, which would have a superelevation appropriate for its design speed. Similarly, the design of freeway and expressway ramps at exits suggests that no penalty should be applied to the mainline, even if the mainline changes direction.

Type Code	Intersection Type	Angle (degrees)	Penalty	Number	Percent
Through	Through	-15 – 15	(none)	13,048	15.6%
Veer Rt	Veer Right	15 – 45	0.1	5,897	7.1%
Right	Right Turn	45 – 120	0.5	13,002	15.6%
Sharp Rt	Sharp Right	120 – 180	0.8	2,761	3.3%
Veer Lft	Veer Left	-15 – -45	0.3	5,361	6.4%
Left	Left Turn	-45 – -120	0.8	12,608	15.1%
Sharp Lft	Sharp Left	-120 – -180	1.2	2,470	3.0%
U turn	U-Turn	-180, 180	2.0	25,820	30.9%
Manual	Illegal or Infeasible Turns	(any)	(prohibited)	220	0.3%
Fwy Cont	Freeway Continuation	(any)	(none)	1,210	1.5%
Exp Cont	Expressway Continuation	(any)	(none)	896	1.1%
Man OCC	Origin Centroid Connector	(any)	(prohibited)	119	0.1%
Man DCC	Destination Centroid Connector	(any)	(prohibited)	48	0.1%
Totals				**83,460**	**100.0%**

Notes: Turn penalties values are in the same units as the impedance variable, that is, the link variable to be minimized when constructing the shortest path.

U-turns were identified by the inbound and outbound links having the same identifier, rather than the angle between the two links.

Origin Centroid Connector penalties were used to constrain the first physical link used in each path, by prohibiting turns from the centroid connector to other real links. Similarly, Destination Centroid Connector penalties were used to constrain the last physical link used in path-building. See sections 3.1.2 for further details.

Percentages do not add to 100 percent because of rounding.

Table 4-6. Types of Turn Penalties.

We wish to have a label reflecting the natural tendency of travelers to select routes such that one goes “up the hierarchy” of roads from local streets to arterials to expressways and freeways during the (roughly) first half of the journey, and then descends “back down the hierarchy” to reach the destination. Ben-Akiva, Bergman, Daly and Ramaswamy (1984) use different weights on the time spent on facilities of different hierarchies to model this label. This was also our initial approach. However, this technique does not adequately capture the importance of sequencing in developing a hierarchical path. Consider two very different paths made up of links of equal length: Path A uses five arterial links, followed by 10 freeway links, and then five more arterial links. Path B also has 10 arterial and freeway links each, but alternates one freeway link after each arterial link. Using the weighting method produces the same impedance for these two paths!

We developed a procedure to use turn penalties to produce more reasonable – and more truly hierarchical – paths. For every link-to-link movement in the network, we consider the hierarchy level of each link. Movements involving no hierarchy change incur no additional penalty beyond the directional penalty described above. Each step up or down the hierarchy results in a minor penalty being added. Further, taking two steps up or down the hierarchy in one movement (for example, going from a local street to an expressway, or from a freeway to a major arterial) is given a greater penalty than two single-step movements. The penalty values used in the hierarchy-based turn penalty file are described in Table 4-7. Table 4-8 describes the structure of the master turn penalty file, which includes fields for the directional penalty only and for the directional plus hierarchy penalty.

	To Hierarchy			
From Hierarchy	1 Freeway	2 Expressway	3 Major Arterial	4 Local Street
1 Freeway	No Penalty	1.50	3.50 + Turn Direction Penalty	7.50 + Turn Direction Penalty
2 Expressway	1.50	No Penalty	1.50 + Turn Direction Penalty	3.50 + Turn Direction Penalty
3 Major Arterial	3.50 + Turn Direction Penalty	1.50 + Turn Direction Penalty	Turn Direction Penalty Only	1.50 + Turn Direction Penalty
4 Local Street	7.50 + Turn Direction Penalty	3.50 + Turn Direction Penalty	1.50 + Turn Direction Penalty	Turn Direction Penalty Only

Notes: Turn penalties values are in the same units as the impedance variable, that is, the link variable to be minimized when constructing the shortest path.
Turn direction penalties are described in Table 4-6.

Table 4-7. Matrix of Penalties for Hierarchical Algorithm.

Name	Range	Description
FROM	1 to 38376	ID of link from which vehicle enters turn
TO	1 to 38376	ID of link to which vehicle completes turn
COMB_ID	100,001.0 to 3,837,638,376.1	Turn penalty ID constructed as 100,000*FROM + TO + 0.1*NONDUPE
FROM_HN	1 to 4, 8	Hierarchy Number of the "from" link (see Table 3-5 for list of codes)
TO_HN	1 to 4, 8	Hierarchy Number of the "to" link (see Table 3-5)
PENALTY	Missing, 0-2	Turn penalty in impedance units; missing value indicates turn prohibition (see Table 4-6)
HN_PEN	Missing, 0-9.5	Turn penalty including transition between link hierarchies penalty (see Table 4-7)
TYPE	See Table 4-6	Type of turn penalty based on direction and function
COMMENT	(character string)	Description of facility/ies associated with turn penalty
DEL_AZI	-180 to 180	Change in link "azimuth" indicating severity of turn. 0 indicates continuing in the same direction; -180 a U-turn. Negative numbers indicate left turns and positive values, right turns.
NONDUPE	0 or 1	1, except for the second occurrence of a U-turn penalty, which is labeled 0
NUMBER	1 to 84,360	Sequentially-numbered turn penalty ID

Table 4-8. Variables in the Turn Penalty File.

4.3 Perceived and Engineering Travel Times

Recall that Section 3.1.3 discusses some hypotheses regarding perceived and engineering travel times. To test the relationship between the estimated travel times supplied by CTPS and those reported by survey respondents, we initially examined a linear regression of reported times on engineering times. We also examined whether SOV or HOV travelers were more likely to report travel times that varied from the network estimates. We also investigated the impact of roadway type (freeway, expressway or arterial) on travel time perception.

We found that perceived times are not significantly different from the coded times, though respondents tended to report trip times that were greater than the network estimates. Nor were vehicle occupancy or roadway type found to have a significant effect on travel time perception. The standard deviation of reported times is about 40 percent of the coded route time.

5 Results

This chapter presents the results of the different types and stages of analysis conducted for this thesis. That is, Chapter 3 describes several modules relating to modeling route choice in the context of heterogeneous driver knowledge. These modules are discussed in the following sections. First, we describe the results of our investigation into using different algorithms to generate possible feasible routes. Section 5.1 discusses the coverage and computational properties of the various algorithms, and summarizes the resulting choice set.

Section 5.2 deals with describing network knowledge from socioeconomic variables and responses to an attitudinal survey. Estimates of parameters from a MIMiC model of network knowledge are presented. These parameters are used to calculate fitted values of network knowledge, which are then used during the estimation of route choice models.

The next two sections deal with the estimation of route choice models. Section 5.3 focuses on different Path-Size Logit specifications. Section 5.4 uses the same utility specification to estimate other types of route choice models – C-Logit, Cross-Nested Logit, Logit Kernel and IAP Logit. Properties of these model types are discussed.

In section 5.5, we compare the various model types against each other. The models are evaluated on the basis of goodness-of-fit, ease of use, computational requirements and basis in theory. Section, 5.6, compares results of using these models for forecasting travelers' responses to a variety of scenarios. Finally, section 5.7 summarizes this chapter.

5.1 Evaluation of Choice Set Generation Algorithms

We examined several variations of the four broad types of route generation algorithms described above: labeling, link elimination, link penalty and simulation. Table 5-1 shows the coverage results of individual labels. That is, each algorithm generates exactly one route by minimizing a particular label. In the instances where a label has parameters, such as the trade-off between time and distance, we use the set of parameters producing the greatest coverage.

From Table 5-1, we can note that no single label performs very well. Minimizing free-flow time produces the best results, and even then, less than one-half the respondents appear to choose a minimum-free-flow-time path. Even fewer appear to follow a minimum-distance path. It can further be noted that combining the 16 algorithms presented in Table 5-1 still does not produce a satisfying result, as 15 to 25 percent of observations do not have sufficient overlap with any of the generated routes, depending on the threshold chosen. Therefore, we examine algorithms that generate multiple paths, such as the link elimination and link penalty K-Shortest Path heuristics, and simulation. Results of these algorithms are compared with labeling in Table 5-2.

Just as we “calibrated” the parameters of generalized cost labels to yield the greatest coverage, we also calibrated the distributional parameters used for simulating travel times. We first calculated the standard deviation of drivers’ “perception errors,” that is, the percentage difference between network times and those reported by survey respondents. We found good coverage results when we drew link travel times from a distribution having a standard deviation twice that of driver perception errors.

Table 5-2 shows that the K-Shortest Path heuristics do increase coverage over labeling alone. As expected, the simulation approach shows diminishing returns with respect to the number of draws. At 48 draws, simulation provides better coverage than the three labels that require no parameters: distance, free-flow time and estimated time. However, simulation does not do better than any individual K-Shortest Path heuristics, or the labeling approach with all 16 labels shown in Table 5-1.

Algorithm and Parameters	Overlap Required for Coverage					
	100 %		90 %		80 %	
Least Time	64	34%	69	37%	84	45%
Least Free-Flow Time	63	34%	70	37%	87	46%
Minimize Generalized Cost Minimize 0.4*Time + 0.4*Distance + 0.2*Toll	62	33%	67	36%	77	41%
Minimize V/C-Weighted Time Minimize CC Time + 0.8 Time 1(V/C=0) + Time 1(0<V/C<0.9) + 0.9 Time 1(V/C> =0.9)	61	32%	67	36%	81	43%
Minimize Left Turns Path Double or Triple Left Turn Penalty	58	31%	66	35%	81	43%
Maximize Capacity-Weighted Time Path	55	29%	64	34%	74	39%
Maximize Time in Secure Neighborhoods weighted by median income	55	29%	60	32%	76	40%
Maximize High Capacity Roads Path Min (High Cap + 2 Low Cap + CC) Time	45	24%	50	27%	65	35%
Turn-Penalty Hierarchy Path (1.5 min for one level higher or lower)	42	22%	49	26%	63	34%
Maximize Freeways Path Minimize (Fwy + 2 Exp + 4 Art + CC) Time	38	20%	46	24%	56	30%
Least Distance	38	20%	42	22%	53	28%
Minimize Number of Links	33	18%	55	29%	57	30%
Maximize Expressways Path Minimize (2 Fwy + Exp + 2 Art + CC) Time	33	18%	34	18%	43	23%
Maximize Arterials Path Minimize (4 Fwy + 2 Exp + Art + CC) Time	27	14%	27	14%	30	16%
Minimize Tolls (and Turn Penalties)	18	10%	19	10%	28	15%
Minimize Stop Lights (Number of Non-Fwy, Non-Expwy Links)	15	8%	17	9%	26	14%
Total of All Above Algorithms	136	72%	143	76%	160	85%

Notes: 188 observations total. Algorithms are sorted in descending order of coverage at the 100 percent overlap threshold.

Table 5-1. Coverage of Individual Single-Route Generation Algorithms for Boston.

	Overlap Required for Coverage					
Algorithm and Parameters	**100 %**		**90 %**		**80 %**	
Total of All Labeling Algorithms (16 Labels)	136	72%	143	76%	160	85%
Total of Minimize Distance, Free-Flow Time and Time	74	39%	82	44%	97	52%
K-Shortest Paths – Link Penalty / De La Barra 40 Unique Routes	102	57%	120	67%	143	80%
K-Shortest Paths – Link Penalty / De La Barra 15 Unique Routes	101	56%	118	66%	139	78%
K-Shortest Paths – Link Elimination / DynaMIT (Includes Least Time Matches)	113	60%	119	63%	134	71%
Total of All Above Algorithms ("Deterministic" Link Attributes)	156	83%	164	87%	175	93%
Minimize Simulated Time 48 Draws	94	50%	120	64%	148	79%
Minimize Simulated Time 32 Draws	92	49%	115	61%	143	76%
Minimize Simulated Time 16 Draws	82	44%	106	56%	133	71%
Minimize Simulated Time 8 Draws	71	38%	95	51%	121	64%
Total of All Above Algorithms	157	84%	165	88%	177	94%

Notes: 188 observations total. Algorithms are sorted by type, and then in descending order of coverage at the 100 percent overlap threshold.

Table 5-2. Coverage of Multiple-Route Generation Algorithms for Boston.

In evaluating route choice generation algorithms, we also need to consider computational performance. An algorithm that yields a five percent increase in the number of observations covered may not be cost-effective if it takes months to run, for example. The results of computational time experiments are shown in Table 5-3 below. Minimizing one label is the fastest, as this simply requires a call to the built-in shortest-path routine. Minimizing a random draw is almost as fast; time must be allowed to make the draws of random travel time before constructing the shortest paths. The link elimination and link penalty heuristics, which involve multiple shortest-path calls, take successively longer.

The computational times presented are based on a GIST platform, as GIST use is prevalent among MPOs. Therefore, the results shown may be affected by the GIST file structure. The link penalty approach seems to perform particularly poorly because updating the costs on a few links requires re-writing the whole network database. In comparison, the link elimination heuristic can be fairly efficiently implemented – a "link in use" bit can be turned on or off. Other GIST or dedicated transportation planning software may produce different results. Our goal was to compare path generation algorithms and not evaluate the relative speed of various transportation planning software packages.

Algorithm Description	**Time for 1 OD Pair**	**Time for 188 OD Pairs**
Minimize One Label	32 s*	1 h 40 min
Minimize a Random Draw	35 s*	1 h 50 min
Minimize 48 Random Draws	3 min 20 s*	10 h 30 min
Link Elimination (DynaMIT)	7 min	22 h*
Link Penalty (De la Barra) for 15 Unique Routes	25 min	3 d 6 h*
Link Penalty (De la Barra) for 40 Unique Routes	1 h 40 min	13 d*

Notes: * indicates a calculated quantity.
Computational experiments were conducted using TransCAD 3.1 on a 400 MHz Pentium II workstation with 256 MB RAM running Windows NT 4.0.

Table 5-3. Computational Times of Route Generation Algorithms

It is also interesting to note that there are some non-linearities in algorithm performance. For example, minimizing 48 random draws does not take as long as minimizing one draw. This may be caused by the fixed computational overhead involved with loading the network database into memory. In contrast, the link penalty algorithm to generate 40 paths takes about four times as long as when generating only 15 paths. This is not surprising considering the structure of the algorithm. When the algorithm starts, it may not be difficult to find many unique paths after increasing the impedance on certain links. However, network topology limits the number of paths having a total impedance of a certain percent greater than that of the shortest path. Therefore, the link penalty algorithm may spend many iterations identifying paths that were previously found and increasing impedance to find new paths. Further, the link penalty implementation was highly memory- and disk-intensive, and workstation performance suffered as memory resources became scarce.

The long computational times of the link penalty approach disqualified it from further consideration. We also had reservations about the realism of paths generated by the link elimination approach. Since we eliminated only one link at a time, it was feared the other generated paths would closely resemble the original shortest path, with the exception of a brief deviation. We were pleased with the computational time of the simulation algorithm, and its ease of implementation. By considering both coverage and computational time, we decided to use simulation with 48 draws and labeling with the three parameter-free objective functions for our "final" choice set generation. Other labels produced paths similar to those from minimizing distance, free-flow time or estimated time. Further, it was not clear that the path generation parameters would be transferable to other areas, or that other even more exotic attributes would be available.

Figure 5-1 presents the coverage of the final choice set generation procedure graphically. The figure may be thought of as similar to a cumulative distribution function of a random variable. By choosing an overlap threshold percentage along the horizontal axis – note that 100 percent overlap is at the left! – one can read off the percent of observations covered using that overlap threshold on the vertical axis. For example, and comparison with Table 5-2, 106 observations, or 56 percent, meet the 100 percent overlap threshold. That is, 106 observations are replicated link-for-link. Considering other thresholds, 134, or 71 percent, have 90 percent overlap or better; and 160, or 85 percent, have at least 80 percent overlap. We chose to use an 80 percent overlap threshold for the estimation set because of the difficulty in generating realistic routes and to have more observations from which to develop the choice models.

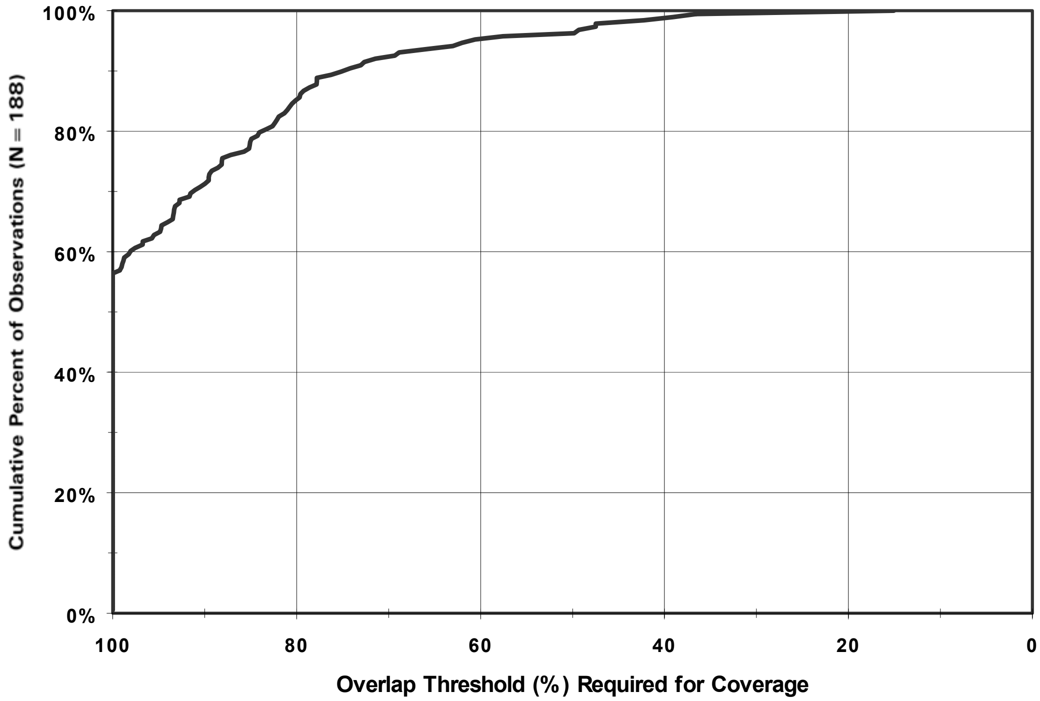


Figure 5-1. Distribution of Coverage.

We examined the observations that failed to make the 80 percent overlap threshold, to see if they might reveal any insights for other path generation algorithms. For instance, we checked whether these observations tended not to overlap more during the collection (access), “line-haul,” or distribution (egress) parts of the trip. We found no such patterns. In fact, our examination revealed that classifying some links as “line-haul” may be an over-simplification. For a given O-D pair, there may be many natural line-haul corridors. However, these are connected by many opportunities to change corridor – for example, bridges between Memorial and Storrow Drives, or cross-streets in general. We found that these observations with lower overlaps tended to change corridors at places not used by the path generation algorithms. That is, while all the links on the observed path may be "covered" by some algorithm, the problem was that all observed links weren't used by the same algorithm, thus leading to larger overlap. This does suggest some future directions for path generation algorithms, which will be discussed in Chapter 6.

Note that the "final" choice set generation procedure would generate up to 51 alternative routes (from three deterministic labels and 48 random draws). Some origin-destination pairs would have fewer alternatives available, as some labels or draws might yield duplicate paths. The distribution of the number of unique paths in the choice sets of the 188 auto users in our sample is shown in Figure 5-2. It can be seen that the median size of the generated choice set is about 30 routes, and that about one-quarter of the observations have a choice set with 40 or more feasible routes.

Also notice that some observations have a very small choice set. These observations generally correspond to employees who live close to the MIT campus. The density of streets in the network is such that these people have few reasonable alternative routes to MIT. That is, the nearest parallel facility may be quite far from the best route, when considered in relation to the total distance between origin and destination. For example, one of the 160 observations having at least 80 percent overlap has only one route available. This respondent lives near Central Square, Cambridge, and uses Mass. Ave. to come to MIT. Broadway Street is the next closest alternative, but this route was sufficiently longer that it wasn't selected as any of the 51 generated paths. Because this individual's choice set represents a captivity situation, we drop it from the data set for estimating route choice models. This leaves 159 observations in the estimation data set.

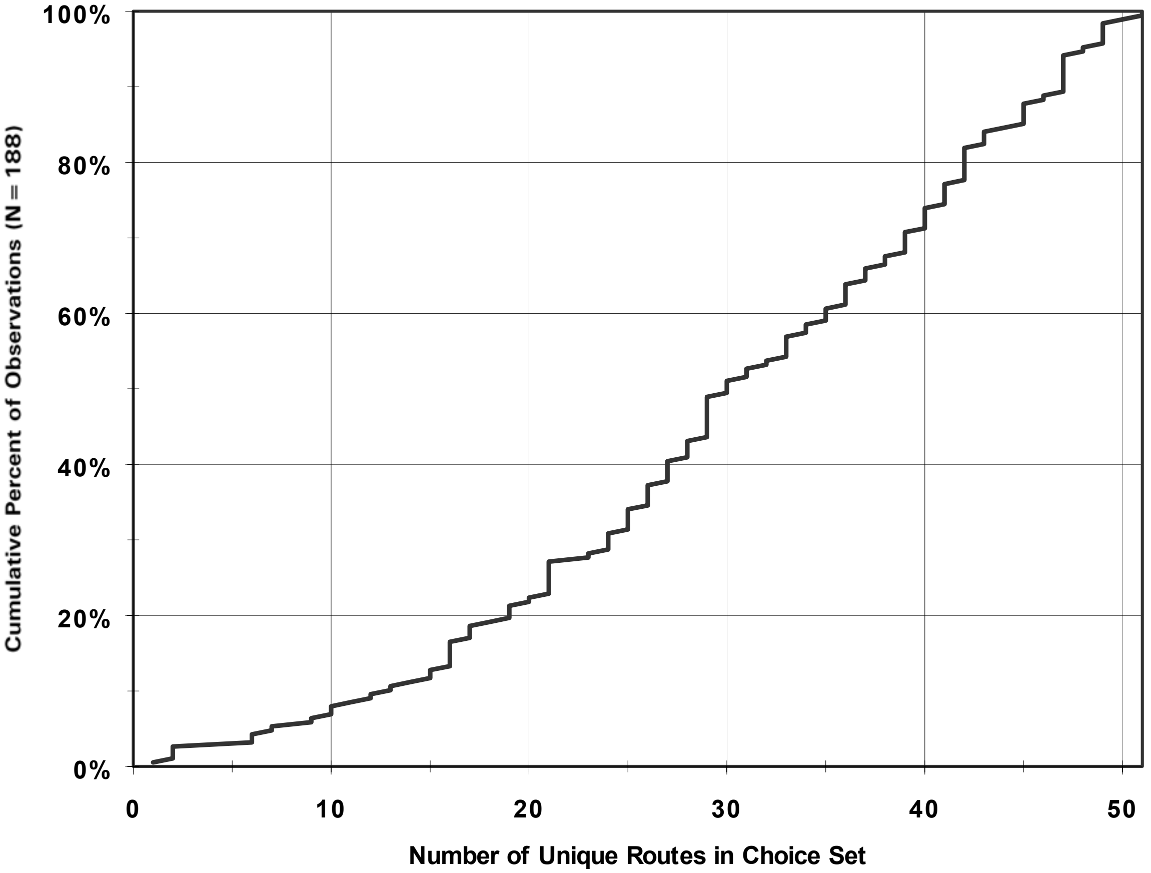


Figure 5-2. Distribution of the Number of Alternative Routes Generated.

The distribution of the choice set size for members of the estimation data set is shown in Figure 5-3. Note this distribution has a similar shape as that for all 188 observations.

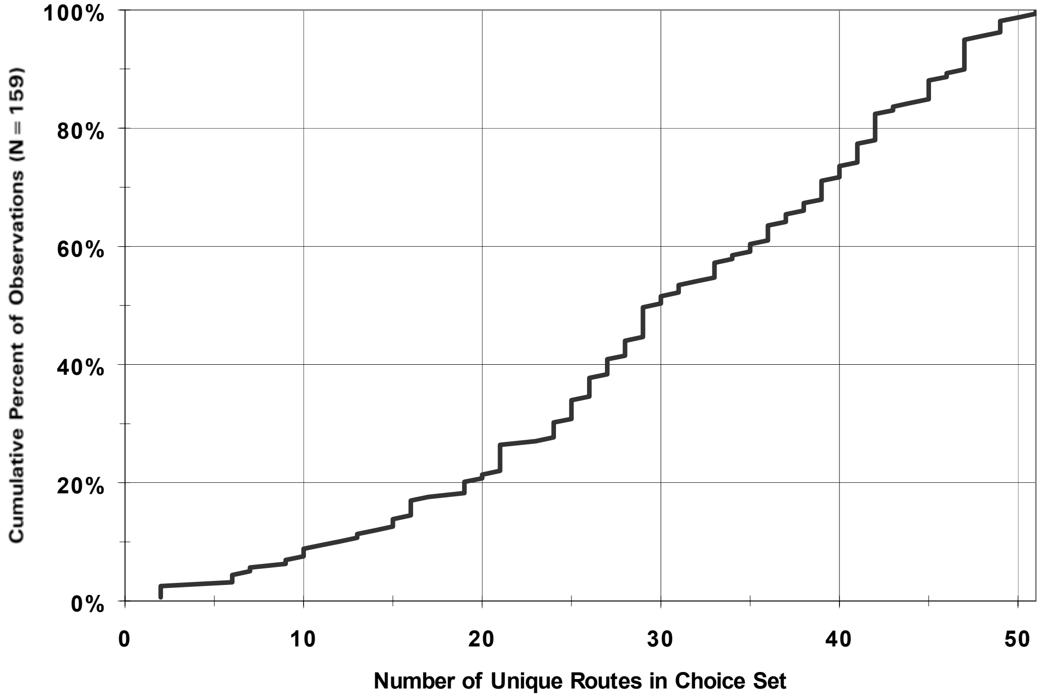


Figure 5-3. Distribution of Number of Alternative Routes for Estimation Set.

Figure 5-4 shows the distribution of the number of links in the choice sets of the 159 respondents used to estimate route choice models. The number of links is an important statistic because it determines the complexity of the Cross-Nested Logit and Logit Kernel models. In the CNL model, the number of nests is equal to the number of links. In Logit Kernel, each link corresponds to a Gaussian error term. The observation using the greatest number of links has 856 links in its choice set. (Note that if we were estimating a Probit or Logit Kernel model using a numerical approximation of the Gaussian c.d.f., this would require solving an 856-dimensional integral.) The smallest choice set among the 159 respondents has 19 links.

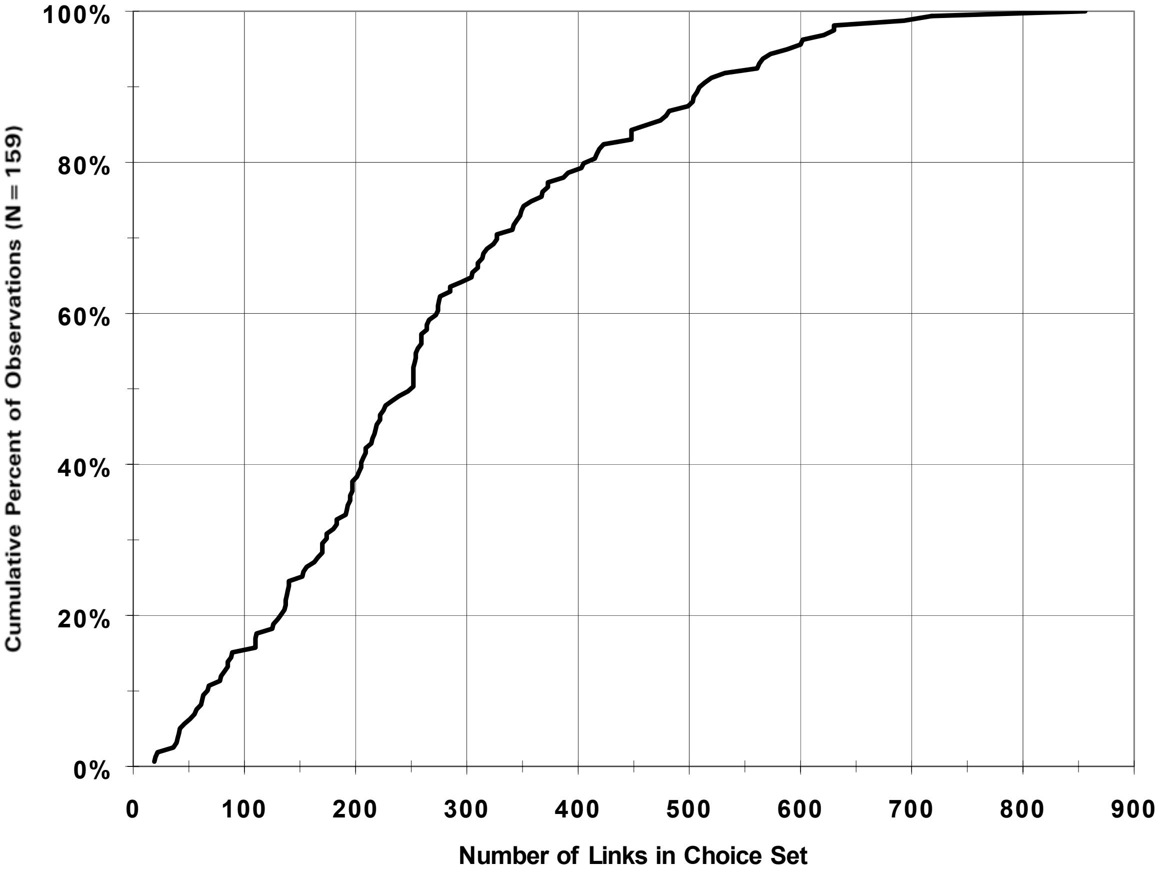


Figure 5-4. Distribution of Number of Links Used in Estimation Set.

Finally, it is worth noting that the large variation in the number of links in the choice set has important implications for programming the estimation procedure. The analyst must make a trade-off between using the same (maximum) number of links per observation or writing more complicated code to adapt to different choice set sizes. If the number of links is held constant across observations, there will be many zeros for some observations. This implementation may result in inefficient use of memory or disk storage space. Of course, coding to first input the number of links in each observation's choice set, and checking that the right number of link data are read in represents a significant effort on the part of the programmer. Note that the CNL program presented in Appendix section D.2 takes the constant-size approach. (The LK estimation code, which is not discussed in this thesis, uses the adaptive technique.)

5.2 MIMiC Model of Network Knowledge

The responses of employees also suggested that familiarity affects the variety of commute route, as shown in Table 5-4. Of the employees disagreeing with the statement, “I know my way around the city well, and can easily find another route,” 57 percent use the same route to MIT every day. In contrast, 60 percent of those agreeing with the statement change their route, and 18 percent of those agreeing change route more often than one workday out of five.

We conducted exploratory factor analysis of respondents’ attitudes toward navigation. This helped us develop a Multiple Indicator-Multiple Cause (MIMiC) specification to examine how network knowledge may be related to a person’s tenure in the metropolitan area and other socio-economic factors. The most reasonable estimation results are shown in Figure 5-5. Note that self-reported learning from maps or experience are significant determinants of spatial knowledge. After controlling for these causes, the relationship between tenure in Boston and spatial knowledge is not significant. It is possible that longer tenure in Boston would lead to more experience, and travelers may be more likely to have learned a route from maps after also spending a few years in Boston. However, our results are consistent with Freundschuh (1992), who was not able to establish a relationship between spatial knowledge and years of exposure, and Stern and Leiser (1988), who concluded that most residents stop learning new routes after a few years.

Also note that we were not able to find any significant relationship between gender and spatial knowledge. This result is similar to that of Lawton, Charleston and Zieles (1996), which investigates indoor navigation. In that study, women and men exhibit similar patterns of route choice, but women are more likely to report uncertainty and to explore the unfamiliar experimental environment, the basement of a university building.

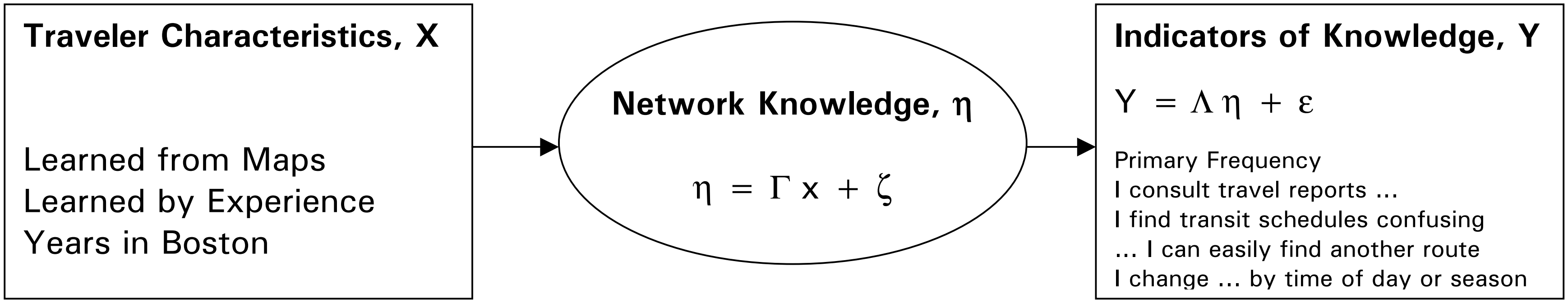
Percentage of trips to MIT by primary mode and route	To what extent do you agree with the statement “I know my way around the city well, and can easily find another route?							
	Agree		Neutral		Disagree		Total	
100%	318	(67%)	82	(17%)	75	(16%)	475	(100%)
	(40%)	(29%)	(50%)	(8%)	(57%)	(7%)	(44%)	(44%)
80 to 99 %	331	(76%)	61	(14%)	46	(10%)	438	(100%)
	(42%)	(30%)	(37%)	(6%)	(35%)	(4%)	(40%)	(40%)
50 to 80 %	116	(83%)	15	(11%)	9	(6%)	140	(100%)
	(15%)	(11%)	(9%)	(1%)	(7%)	(1%)	(13%)	(13%)
Less than 50%	27	(73%)	7	(19%)	3	(8%)	37	(100%)
	(3%)	(3%)	(4%)	(1%)	(1%)	(<1%)	(3%)	(3%)
Total	792	(73%)	165	(15%)	133	(12%)	1090	(100%)
	(100%)	(73%)	(100%)	(15%)	(100%)	(12%)	(100%)	(100%)

Key	Count	(Row%)
	(Col%)	(Cell%)

Note: 1090 respondents answering both questions.

Source: MIT Planning Office, 1997.

Table 5-4. Relationship Between Familiarity and Variation in Commute Patterns.



Traveler Characteristics, X

Learned from Maps
Learned by Experience
Years in Boston

Network Knowledge, η

$\eta = \Gamma x + \zeta$

Indicators of Knowledge, Y

$Y = \Lambda \eta + \varepsilon$

Primary Frequency
I consult travel reports …
I find transit schedules confusing
… I can easily find another route
I change … by time of day or season

Cause	Gamma	t-stat
Learned from Maps	0.15	2.97
Learned from Experience	0.18	3.36
Years in Boston	0.0012	0.97

Indicator	Lambda	t-stat
Primary Frequency (1 = change often; 5 = same way all the time)	-1.3	-4.25
I consult travel reports … (1 = disagree strongly; 5 = agree strongly)	0.59	1.87
I find transit schedules confusing	-0.13	-0.59
… I can easily find another route	1	*
I change how I travel by time of day or season of year	3.0	3.95

Note: * This parameter is normalized to unity for identification.

Figure 5-5. MIMiC Model Structure and Estimated Coefficients.

There is a particularly strong relationship between spatial knowledge and two indicators: the frequency the primary route is used, and the tendency to vary route by season or time of day. Greater spatial knowledge suggests greater variation in route, as expected. The relation with the desire to consult travel reports is marginally significant, which would support the hypothesis that those travelers who are aware of more routes might seek out information to decide among them (or alternatively, that listening to travel reports may expose one to new routes). The relationship with aptitude for using transit schedules is not significant, but is retained since it has the expected sign.

Because the means by which people learn about routes is not a readily-available demographic variable (e.g., available in the Census), forecasting from this model would require either sample enumeration or development of models that predict learning from more readily-available variables.

5.3 PS Logit Route Choice Models

Table 5-5 shows the coefficient estimates from several Path Size Logit specifications. We examined various specifications including piece-wise linear specifications for both free-flow time and delay components, travel time by facility classes, the interaction of time spent in insecure neighborhoods with income and gender, and the interaction of time spent on numbered routes with gender and staff classification. The four most promising models are presented in Table 5-5. These models use a Path Size term calculated where link "lengths" are based on free-flow time rather than distance, and the parameter γ in the Exponential Path Size formulation is infinity. This choice of Path Size formulation will be discussed later.

The four models presented were selected because they have the appropriate sign on all coefficients, and the coefficient estimates are statistically significant or there is a compelling reason for their inclusion. For example, the coefficients for the Mass. Pike and Sumner Tunnel dummies are not significant. However, these dummies act in a manner similar to that of

Coefficient	Model 1	Model 2	Model 3	Model 4
Distance **Estimate** (Std. Error) T-Statistic	**-0.181** (0.101) -1.8	**-0.178** (0.107) -1.7	**-0.205** (0.105) -1.9	**-0.212** (0.100) -2.1
Free-Flow Time	**-0.506** (0.874) -0.6			**-0.513** (0.082) -6.3
Free-Flow Time between 0 and 15 min		**-0.563** (0.243) -2.3	**-0.659** (0.255) -2.6	
Free-Flow Time between 15 and 45 min		**-0.527** (0.100) -5.3	**-0.512** (0.093) -5.5	
Free-Flow Time in excess of 45 min		**-0.502** (0.189) -2.7	**-0.489** (0.174) -2.8	
Mass. Pike Dummy	**-0.70** (0.60) -1.2	**-0.67** (0.61) -1.1	**-0.51** (0.60) -0.8	**-0.49** (0.60) -0.8
Tobin Bridge Dummy	**2.53** (0.90) 2.8	**2.50** (0.95) 2.6	**2.65** (0.93) 2.9	**2.75** (0.88) 3.1
Sumner Tunnel Dummy	**2.13** (1.16) 1.8	**2.00** (1.18) 1.7	**1.89** (1.16) 1.6	**1.92** (1.15) 1.7
Delay Est. Time – Free-Flow Time	**-0.261** (0.132) -2.0			
Delay for No Income Reported		**-0.640** (0.295) -2.2		
Delay for Income less than $100,000 per year		**-0.209** (0.143) -1.5		

Table 5-5. Estimates from Different PSL Specifications.

Coefficient	Model 1	Model 2	Model 3	Model 4
Delay for Income $100,000 to $150,000 per year		**-0.290** (0.253) -1.1		
Delay for Income $150,000 or more per year		**-0.358** (0.420) -0.9		
ln(Delay) for No Income Reported			**-4.45** (1.81) -2.5	**-4.45** (1.77) -2.5
ln(Delay) for Income less than $100,000 per year			**-0.662** (0.433) -1.5	**-0.583** (0.410) -1.4
ln(Delay) for Income $100,000 or more per year			**-2.739** (0.907) -3.0	**-2.676** (0.895) -3.0
Time on Numbered Routes	**0.094** (0.032) 2.9	**0.095** (0.032) 2.9	**0.090** (0.031) 2.9	**0.090** (0.031) 2.9
Least Distance Path Dummy	**0.765** (0.248) 3.1	**0.770** (0.254) 3.0	**0.737** (0.254) 2.9	**0.759** (0.249) 3.0
Least Estimated Time Path Dummy	**0.375** (0.255) 1.5	**0.360** (0.259) 1.4	**0.342** (0.251) 1.4	**0.377** (0.244) 1.5
ln(Path Size) **Estimate** Based on Free-Flow (Std. Error) Time T-Stat w/r/t 0 $\gamma = \infty$ T-Stat w/r/t 1	**0.762** (0.124) 6.1 -1.9	**0.746** (0.125) 6.0 -2.0	**0.732** (0.112) 6.5 -2.4	**0.730** (0.122) 6.0 -2.2
Number of Coefficients	**10**	**15**	**14**	**12**
Log-Likelihood at Estimates	**-398.4**	**-397.2**	**-392.9**	**-393.1**
Rho-bar-squared	**0.214**	**0.207**	**0.217**	**0.221**

Notes: N = 159. Log-likelihood for all coefficients at zero is –519.70.

Table 5-5. Estimates from Different PSL Specifications (Continued).

alternative-specific constants. Since omitting alternative-specific constants can result in bias to other parameters, we retain the facility dummies. Similarly, we retain the least estimated travel time path dummy. This dummy appears to be correlated with the Path Size term. In specifications where the least estimated time path dummy was omitted, the coefficient on Path Size became insignificantly different from zero. Finally, some income interactions with delay were not strongly significant. These were retained so that delay remained in the utility function for all income groups. The lack of significance may be related to small numbers of respondents in a particular income group.

For the piece-wise linear specifications of free-flow time (Models 2 and 3), the utility coefficient decreases in absolute value as total free-flow time increases. This is reasonable, as someone with a longer commute would be less sensitive to the marginal impact of an additional minute of travel. The free-flow time coefficient estimates are not significantly different, as can be seen by comparing the log-likelihoods of Models 3 and 4. This result may arise from the limited number of observations in the data set. We expect that a larger data set would have more explanatory power to estimate a piece-wise linear travel time specification.

The Mass. Pike dummy coefficient is negative and equivalent to about a minute of free-flow time. The Tobin Bridge and Sumner Tunnel dummies are positive, and equivalent to about four or five minutes of free-flow time. We interpret this as the Tobin Bridge and Sumner Tunnel being very prominent facilities for crossing a major barrier – the Boston Harbor and Mystic River. Travelers coming from the north east of MIT might use these prominent features in their trip planning. The Mass. Pike, being a linear feature rather than a barrier crossing, may not have the same prominence. Its negative coefficient may be more related to delays experienced at toll booths.

Models 2, 3 and 4 suggest that individuals with higher household incomes are more sensitive to delays, which is reasonable from value-of-time considerations. Also note that the coefficient on delay (or its log) for those subjects who did not provide their income is larger in magnitude than the coefficient for the subjects checking the highest income categories. The following hypothesis is consistent with this result: travelers with the highest household income would be most sensitive to delay, and would also be most reluctant to disclose their income on a survey. We examined piece-wise linear specifications of delay, and their interactions with income. While the delay coefficients did diminish with increased delay – as was also noticed with free-flow time – the data did not appear to support such fine classification. That is, with so many coefficients on the interaction of income and delay pieces, very few were statistically significant. The logarithmic specification of delay also captures diminishing disutility and produces more significant coefficients.

The coefficient on time spent on numbered routes is positive and about one-fifth the magnitude of the free-flow time coefficient. This is consistent with numbered routes having a prominence effect – that is, the subject's mental maps are more likely to consist of numbered routes. The label dummies are also positive, which is consistent with those paths gaining prominence.

Discrete choice theory suggests that the Path Size coefficient should have a value of one. Since we do not know the size of paths with certainty and wish to examine different Path Size specifications, we include a coefficient on the log of Path Size during the estimation results. The estimates for the four models are remarkably stable. The Path Size coefficient is significantly different from zero, and the t-statistic near 2 suggests that the coefficient is also significantly different from one. While the Path Size coefficient is closer to one than zero, we would hope it would be much more so. The value of the Path Size coefficient being less than one indicates that the Path Size correction may be more severe than necessary to account for travelers' perceptions of correlation across routes. Since the Path Size term is correlated with the explanatory variables (particularly free-flow time, numbered route time and the label dummies), it may be that some of these variables are accounting for overlapping, so the full effect of the Path Size correction is not necessary.

Since the four models presented in Table 5-5 all have coefficients with reasonable signs and magnitudes, we turn to measures of goodness of fit to select the preferred model of the four. The rho-bar-squared statistic favors a parsimonious specification; that is, additional explanatory variables are expected to significantly improve the overall model log-likelihood. Based on this criterion, we choose Model 4 as our preferred or "reference" specification. All future comparisons of various model types and their parameter specifications will use this set of explanatory variables.

We now turn to the discussion of calibrating the Path Size parameter. Since the Path Size term is based on network topology, we found it most efficient to calculate the Path Sizes within TransCAD, and estimate PSL models with ALOGIT. (Any other MNL software would also work.) This led to a "calibration" approach: we produced several data files, each based on a different value of the Path Size parameter, γ. The reference model specification (that is, the set of X's) was estimated for each data set. These results are presented in Table 5-6 below. Note that the log-likelihood and rho-bar-squared are monotonically increasing in γ. As discussed in Section 4.1.3, low values of γ may lead to counter-intuitive results, so we did not expect good fit from the models estimated on these data sets. However, notice that there is a "jump" of about four log-likelihood points when we go from a gamma of 99 to infinity. This result is surprising, because it suggests the Path Size term is based on a deterministic label. That is, a path that overlaps with one that is infinitesimally shorter gets absolutely no link size (and therefore a substantially reduced path size) for the overlapping segment. An empirical consideration is that the Path Size label (here, free-flow time) must be known with certainty, or specifications with Path Sizes based on different labels must be tested. We remain somewhat skeptical of this result, and would encourage other researchers to use a large finite value of gamma should calibrating the Path Size formulation be prohibitive.

Coefficient		$\gamma = 0$	$\gamma = 1$	$\gamma = 4$	$\gamma = 99$	$\gamma = \infty$
Distance	**Estimate**	**-0.256**	**-0.256**	**-0.248**	**-0.204**	**-0.212**
	(Std. Error)	(0.106)	(0.106)	(0.106)	(0.100)	(0.100)
	T-Statistic	-2.4	-2.4	-2.3	-2.0	-2.1
Free-Flow Time		**-0.664**	**-0.669**	**-0.653**	**-0.492**	**-0.513**
		(0.106)	(0.104)	(0.098)	(0.080)	(0.082)
		-6.3	-6.4	-6.7	-6.1	-6.3
Mass. Pike Dummy		**-0.80**	**-0.83**	**-0.76**	**-0.52**	**-0.49**
		(0.68)	(0.68)	(0.67)	(0.58)	(0.60)
		-1.2	-1.2	-1.1	-0.9	-0.8
Tobin Bridge Dummy		**3.08**	**3.10**	**3.04**	**2.76**	**2.75**
		(0.95)	(0.95)	(0.94)	(0.88)	(0.88)
		3.3	3.3	3.2	3.1	3.1
Sumner Tunnel Dummy		**1.93**	**1.89**	**2.09**	**1.73**	**1.92**
		(1.20)	(1.20)	(1.20)	(1.13)	(1.15)
		1.6	1.6	1.7	1.5	1.7
ln(Delay) for No Income Reported		**-5.25**	**-5.26**	**-5.25**	**-4.26**	**-4.45**
		(1.93)	(1.93)	(1.92)	(1.72)	(1.77)
		-2.7	-2.7	-2.7	-2.5	-2.5
ln(Delay) for Income less than $100,000 per year		**-0.328**	**-0.348**	**-0.325**	**-0.614**	**-0.583**
		(0.445)	(0.445)	(0.441)	(0.416)	(0.410)
		-0.7	-0.8	-0.7	-1.5	-1.4
ln(Delay) for Income $100,000 or more per year		**-2.697**	**-2.719**	**-2.658**	**-2.736**	**-2.676**
		(0.947)	(0.945)	(0.941)	(0.895)	(0.895)
		-2.9	-2.9	-2.8	-3.1	-3.0
Time on Numbered Routes		**0.110**	**0.110**	**0.113**	**0.089**	**0.090**
		(0.032)	(0.032)	(0.032)	(0.031)	(0.031)
		3.4	3.4	3.5	2.9	2.9
Least Distance Path Dummy		**0.994**	**0.982**	**0.993**	**0.725**	**0.759**
		(0.257)	(0.257)	(0.255)	(0.254)	(0.249)
		3.9	3.8	3.9	2.9	3.0
Least Estimated Time Path Dummy		**0.961**	**0.958**	**0.956**	**0.466**	**0.377**
		(0.227)	(0.227)	(0.227)	(0.244)	(0.244)
		4.2	4.2	4.2	1.9	1.5
ln(Path Size)	**Estimate**	**0.497**	**0.567**	**0.498**	**0.933**	**0.730**
Exponential Specification	(Std. Error)	(0.420)	(0.419)	(0.336)	(0.128)	(0.122)
Based on Free-Flow Time	T-Stat w/r/t 0	1.2	1.4	1.5	5.1	6.0
	T-Stat w/r/t 1	-1.2	-1.0	-1.5	-0.5	-2.2
Log-Likelihood at Estimates		**-410.1**	**-409.9**	**-409.7**	**-397.7**	**-393.1**
Rho-bar-squared		**0.188**	**0.188**	**0.189**	**0.212**	**0.221**

Notes: N = 159. K (number of coefficients) = 12. Log-likelihood for all coefficients at zero is -519.7.

Table 5-6. Comparison of Different Path Size Specifications.

In Table 5-7, we show that Path Size calculated with the free-flow time label has a better model fit than Path Size based on distance. (Though results are not shown here, Path Size based on free-flow time consistently had better fit than Path Size based on distance for other values of gamma.) The table also shows that either Path Size formulation out-performs plain MNL. Note that many parameters have stable or robust estimates and standard errors across the three specifications. Exceptions to this general observation can be explained by the presence of the Path Size term. When Path Size based on distance is used, the utility coefficients of distance and the least distance path dummy decrease in value and lose significance. Similarly, when Path Size based on free-flow time is used, the free-flow time and least estimated time (that is, free-flow time plus delay) path dummy coefficients become less significant. In the MNL specification, the two label dummies have higher t-statistics than either PSL specification. An interpretation of this result is that in MNL, the label dummies are attempting to explain the prominence of the shortest paths and the perception of those paths by the travelers. With PSL, that perception is built into the Path Size term. The Path Sizes also have greater variation – the label dummies are zero for all but one path, except for the rare case of ties. Therefore, the PSL models provide greater explanatory power. The standard error of the Path Size term and overall model fit suggests that Path Size based on free-flow time provides a better explanation of Boston drivers' choices.

We believe this result is also consistent with the Boston driving environment: as a result of political opposition to freeways, there are few freeways in the Boston area, the existing freeways are limited to specific corridors, and all roadways are frequently congested during heavy travel periods. The heterogeneous nature of the Boston roadway network means that drivers wishing to minimize travel times are not necessarily served by taking the shortest-distance path. The least-distance path may involve slower local roads and many stop lights. Drivers seeking to minimize travel times would prefer higher functional type roadways such as freeways and expressways, which would lead them to take more circuitous routes. If drivers are satisfied with these routes, they will have no incentive to seek other routes, and thus their mental maps will be limited to the major freeways and the arterials necessary to access them. In effect, drivers' mental maps are a

Coefficient		MNL	PS based on Distance	PS based on Free-Flow Time
Distance	**Estimate**	**-0.253**	**-0.184**	**-0.212**
	(Std. Error)	(0.106)	(0.107)	(0.100)
	T-Statistic	-2.4	-1.7	-2.1
Free-Flow Time		**-0.601**	**-0.638**	**-0.513**
		(0.091)	(0.091)	(0.082)
		-6.6	-7.0	-6.3
Mass. Pike Dummy		**-0.64**	**-0.81**	**-0.49**
		(0.68)	(0.67)	(0.60)
		-0.9	-1.2	-0.8
Tobin Bridge Dummy		**2.90**	**2.72**	**2.75**
		(0.94)	(0.92)	(0.88)
		3.1	3.0	3.1
Sumner Tunnel Dummy		**2.18**	**1.99**	**1.92**
		(1.20)	(1.19)	(1.15)
		1.8	1.7	1.7
In(Delay) for No Income Reported		**-5.13**	**-5.04**	**-4.45**
		(1.98)	(1.93)	(1.77)
		-2.6	-2.6	-2.5
In(Delay) for Income less than $100,000 per year		**-0.205**	**-0.337**	**-0.583**
		(0.441)	(0.437)	(0.410)
		-0.5	-0.8	-1.4
In(Delay) for Income $100,000 or more per year		**-2.562**	**-2.666**	**-2.676**
		(0.936)	(0.939)	(0.895)
		-2.7	-2.8	-3.0
Time on Numbered Routes		**0.112**	**0.110**	**0.090**
		(0.032)	(0.032)	(0.031)
		3.5	3.4	2.9
Least Distance Path Dummy		**1.056**	**0.662**	**0.759**
		(0.251)	(0.295)	(0.249)
		4.2	2.2	3.0
Least Estimated Time Path Dummy		**0.971**	**0.904**	**0.377**
		(0.227)	(0.228)	(0.244)
		4.3	4.0	1.5
In(Path Size)	**Estimate**		**0.297**	**0.730**
$\gamma = \infty$	(Std. Error)		(0.124)	(0.122)
	T-Stat w/r/t 0		2.4	6.0
	T-Stat w/r/t 1		-5.7	-2.2
Number of Coefficients		**11**	**12**	**12**
Log-Likelihood at Estimates		**-410.8**	**-408.1**	**-393.1**
Rho-bar-squared		**0.188**	**0.192**	**0.221**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70.

Table 5-7. Comparison of MNL and PSL Specifications.

transformation of the physical world based on travel time. Since drivers' mental maps are conformed to travel time, the PSL model with Path Sizes based on free-flow time is expected to have better fit.

5.4 Other Route Choice Models

To assess the usefulness of the Path Size Logit model, we must also examine other route choice model structures. In this section, we consider the C-Logit, Cross-Nested Logit, Logit Kernel, and IAP Logit models. Estimation results from each of these model types are presented, beginning with C-Logit in section 5.4.1. For each model type, we examine models with and without a Path Size term. This may not seem intuitive – one may argue that if for instance, the Cross-Nested Logit model accounts for path overlapping, then why is a Path Size term with the same purpose included? The motivation for this is purely empirical. Just as the Logit Kernel model allows examining whether the data fit a probit or logit structure better, a hybrid model with Path Size allows us to examine whether Path Size or another model fits the data better.

Similarly, IAP Logit may be thought of as an extension to the utilities of any of the model types. Therefore, in section 5.4.4, we present IAP variants of MNL, PSL, CNL and LK.

5.4.1 C-Logit

Table 5-8 displays the estimation results for C-Logit models with two types of commonality factors:

$$CF_{in} = -\beta_0 \ln \sum_{j \in C_n} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right)^{\gamma} \text{, and} \qquad (2\text{-}5)$$

$$CF_{in} = -\beta_0 \ln \left[1 + \sum_{\substack{j \in C_n \\ j \neq i}} \left(\frac{L_{ij}}{\sqrt{L_i L_j}} \right) \left(\frac{L_i - L_{ij}}{L_j - L_{ij}} \right) \right]. \qquad (2\text{-}8)$$

We also examined models for the other two commonality factor definitions:

$$CF_{in} = -\beta_0 \ln \sum_{a \in \Gamma_i} \frac{l_a}{L_i} N_{an} \text{ , and} \tag{2-6}$$

$$CF_{in} = -\beta_0 \sum_{a \in \Gamma_i} \frac{l_a}{L_i} \ln N_{an} \text{ .} \tag{2-7}$$

Note that in all cases, as the number of overlapping paths increases, a path's commonality factor increases while its size decreases. Therefore, the coefficient on the commonality factor is expected to be negative. However, we did not obtain this result for the commonality factor formulations of equations 2-6 and 2-7, which are not presented in Table 5-8. Further note that none of the commonality factors presented in Table 5-8 are statistically significant, and when a Path Size term is added, the sign of the commonality factor term changes to positive. This result suggests – as do the goodness of fit measures – that the Path Size term does a better job of explaining drivers' perceptions of overlapping paths than do the commonality factors. An intuitive explanation is that Path Size "over-explains" a "first-order" effect of overlapping, so CF, which is left to explain minor adjustments, switches from the expected negative sign without PS to a positive sign to compensate for the powerful Path Size term. Finally note that none of the C-Logit models have a better fit than the best PSL model, Model 4 of Table 5-5. (This model is also presented in the final columns of Tables 5-6 and 5-7.) Therefore, we cannot recommend the use of the C-Logit model for large urban networks.

Coefficient		Commonality Factor from Eqn.				PSL only
		2-5	2-5	2-8	2-8	
Distance	**Estimate**	**-0.258**	**-0.201**	**-0.256**	**-0.207**	**-0.212**
	(Std. Error)	(0.106)	(0.101)	(0.104)	(0.101)	(0.100)
	T-Statistic	-2.4	-2.0	-2.5	-2.1	-2.1
Free-Flow Time		**-0.636**	**-0.455**	**-0.615**	**-0.500**	**-0.513**
		(0.101)	(0.098)	(0.089)	(0.084)	(0.082)
		-6.3	-4.6	-6.9	-5.9	-6.3
Mass. Pike Dummy		**-0.76**	**-0.32**	**-0.80**	**-0.40**	**-0.49**
		(0.68)	(0.64)	(0.65)	(0.62)	(0.60)
		-1.1	-0.5	-1.2	-0.7	-0.8
Tobin Bridge Dummy		**3.00**	**2.57**	**2.99**	**2.69**	**2.75**
		(0.95)	(0.90)	(0.93)	(0.89)	(0.88)
		3.2	2.9	3.2	3.0	3.1
Sumner Tunnel Dummy		**2.06**	**2.07**	**1.96**	**2.02**	**1.92**
		(1.20)	(1.16)	(1.19)	(1.16)	(1.15)
		1.7	1.8	1.7	1.7	1.7
ln(Delay) for No Income Reported		**-5.15**	**-4.36**	**-5.10**	**-4.42**	**-4.45**
		(1.94)	(1.80)	(1.90)	(1.79)	(1.77)
		-2.7	-2.4	-2.7	-2.5	-2.5
ln(Delay) for Income less than $100,000 per year		**-0.268**	**-0.519**	**-0.341**	**-0.536**	**-0.583**
		(0.441)	(0.420)	(0.435)	(0.418)	(0.410)
		-0.6	-1.2	-0.8	-1.3	-1.4
ln(Delay) for Income $100,000 or more per year		**-2.660**	**-2.557**	**-2.722**	**-2.612**	**-2.676**
		(0.938)	(0.913)	(0.924)	(0.908)	(0.895)
		-2.8	-2.8	-3.0	-2.9	-3.0
Time on Numbered Routes		**0.111**	**0.090**	**0.109**	**0.090**	**0.090**
		(0.032)	(0.031)	(0.032)	(0.031)	(0.031)
		3.5	2.9	3.4	2.9	2.9
Least Distance Path Dummy		**1.020**	**0.799**	**0.973**	**0.789**	**0.759**
		(0.256)	(0.252)	(0.256)	(0.252)	(0.249)
		4.0	3.2	3.8	3.1	3.0
Least Estimated Time Path Dummy		**0.966**	**0.344**	**0.920**	**0.369**	**0.377**
		(0.227)	(0.246)	(0.228)	(0.245)	(0.244)
		4.3	1.4	4.0	1.5	1.5
Commonality Factor		**-0.458**	**0.658**	**-0.862**	**0.443**	
		(0.597)	(0.627)	(0.528)	(0.596)	
		-0.8	1.1	1.6	0.7	
ln(Path Size)	**Estimate**		**0.744**		**0.771**	**0.730**
Based on Free-Flow Time	(Std. Error)		(0.129)		(0.134)	(0.122)
$\gamma = \infty$	T-Stat w/r/t 0		6.0		5.8	6.0
	T-Stat w/r/t 1		-1.8		-1.7	-2.2
Number of Coefficients		**12**	**13**	**12**	**13**	**12**
Log-Likelihood at Estimates		**-410.5**	**-392.5**	**-409.6**	**-392.8**	**-393.1**
Rho-bar-squared		**0.187**	**0.220**	**0.189**	**0.219**	**0.221**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70.

Table 5-8. C-Logit Estimation Results.

Cascetta, Nuzzolo, Russo and Vitetta (1996) were able to estimate reasonable C-Logit models, and it is worth examining the differences. Those authors worked with a data set of inter-city trips in Italy. In this setting, it is likely that there are not many (reasonable, realistic) alternative routes available, and the authors did not consider many alternative routes in their estimation procedures. In contrast, the data set presented here uses a large-scale urban network with up to around 50 possible alternative routes. Looking again at the commonality factor definitions, one can see that the value of the commonality factor (before taking the log) can never exceed J_n, the number of paths in the choice set. Note that some CF definitions appear structurally similar to the exponential Path Size term. It may be that the commonality factor produces a satisfactory approximation or explanation of drivers' perception of overlapping routes at small values of J_n, but not at larger values.

5.4.2 Cross-Nested Logit

Next we consider Cross-Nested Logit models. As discussed in Section 2.1.11, Vovsha and Bekhor (1998) initially proposed this model with

$$\alpha_{ai} = \frac{l_a}{L_i}\delta_{ai} \ .$$

No specification for μ was given. Ideally, this quantity would be estimated. Papola (2000) estimated some models with constant μ for all nests and did not obtain satisfactory results except when μ = zero, which corresponds to a deterministic lower nest. We have also been unable to obtain satisfactory results with this specification of μ.

Alternatively, nest-specific coefficients, μ_m, may be used in the estimation process. The large number of links in the network (around 34 thousand for Boston) makes it prohibitive to estimate these coefficients directly. Bekhor (2001) proposed the following formulation based on path topology:

$$\mu_m = 1 - \frac{\sum_{j \in C_n} \alpha_{mi}}{\sum_{j \in C_n} \delta_{mi}}. \tag{2-9}$$

The Red-Bus-Blue-Bus Network may provide some intuition to this formulation. Recall that in Nested Logit models, $\mu = 1$ implies that the alternatives in the nest are uncorrelated (that is, we're back to MNL), and for μ approaching zero, the alternatives are near-perfectly correlated. For the overlapping link, μ is zero, meaning deterministic allocation – since the two paths have the same impedance, they split the shares – and here they are perfectly correlated because they're the same physical path.

We can further parameterize the Bekhor formulation for μ as follows:

$$\mu_m = 1 - \left(\frac{\sum_{j \in C_n} \alpha_{mi}}{\sum_{j \in C_n} \delta_{mi}} \right)^{\gamma}, \tag{5-1}$$

$$\mu_m = \left(1 - \frac{\sum_{j \in C_n} \alpha_{mi}}{\sum_{j \in C_n} \delta_{mi}} \right)^{\gamma}, \text{ and} \tag{5-2}$$

$$\mu_m = \left[1 - \left(\frac{\sum_{j \in C_n} \alpha_{mi}}{\sum_{j \in C_n} \delta_{mi}} \right)^{\gamma_1} \right]^{\gamma_2}. \tag{5-3}$$

Since these new formulations involve power transformations, the new values of μ are unchanged when the original (that is, the Equation 2-9) μ is 0 or 1.

Table 5-9 shows the results of the Cross-Nested Logit model estimations. The PSL estimates are shown in the first column for comparison. Notice that all utility parameter (“beta”) estimates and their standard errors are similar for the specifications that include Path Size. Therefore, we focus on measures of model fit, which help us evaluate the usefulness of the Path Size and Cross-Nested specifications.

Notice that the model with only the Path Size term (in the first column of Table 5-9) out-performs the model with only the Cross-Nested Logit structure (in the second column). This is interesting for several reasons. First, the PSL model is more easily estimated than the CNL model; with PSL, the Path Size term can be calculated after the path enumeration step and used in standard MNL estimation software. CNL estimation requires specialized code (for example, a GAUSS program or BIOGEME). Commensurate with its complexity, CNL takes longer computational time to estimate, as will be discussed in Section 5.5, below. Theoretically, the Path Size term may be thought of as an approximation to CNL in much the same way that the - ln 2 term added to the systematic utilities of Red Bus and Blue Bus in the famous example are an approximation to Nested Logit.

Why then does the PSL model out perform the CNL model? One reason may be that the PSL term has been calibrated with a value of γ= infinity. The CNL specifications estimated in Table 5-9 may be more similar to a Path Size term with γ= 1. Note that the log-likelihood for the PSL model in Table 5-6 with γ= 1 is –409.9. Therefore, CNL is an improvement over PSL with a corresponding specification. Also note that CNL has a better fit than the MNL model, which has a log-likelihood of –410.8 (See Table 5-7).

Also note that the models with both CNL and Path Size specifications out-perform both “pure” CNL and PSL. This observation may be related to the need to further calibrate the CNL models, as discussed above.

Coefficient		PSL only	Cross-Nested Logit with μ_m given by Eqn.			
			2-9	2-9	5-1	5-2
Distance	**Estimate**	**-0.212**	**-0.252**	**-0.224**	**-0.222**	**-0.217**
	(Std. Error)	(0.100)	(0.103)	(0.099)	(0.102)	(0.097)
	T-Statistic	-2.1	-2.5	-2.3	-2.2	-2.2
Free-Flow Time		**-0.513**	**-0.553**	**-0.474**	**-0.469**	**-0.460**
		(0.082)	(0.086)	(0.079)	(0.091)	(0.087)
		-6.3	-6.5	-6.0	-5.1	-5.3
Mass. Pike Dummy		**-0.49**	**-0.53**	**-0.37**	**-0.38**	**-0.38**
		(0.60)	(0.65)	(0.59)	(0.58)	(0.57)
		-0.8	-0.8	-0.6	-0.7	-0.7
Tobin Bridge Dummy		**2.75**	**2.79**	**2.75**	**2.72**	**2.66**
		(0.88)	(0.91)	(0.87)	(0.92)	(0.88)
		3.1	3.1	3.2	3.0	3.0
Sumner Tunnel Dummy		**1.92**	**2.06**	**1.92**	**1.89**	**1.84**
		(1.15)	(1.16)	(1.12)	(1.13)	(1.10)
		1.7	1.8	1.7	1.7	1.7
ln(Delay) for No Income Reported		**-4.45**	**-4.80**	**-4.26**	**-4.21**	**-4.11**
		(1.77)	(1.85)	(1.65)	(1.70)	(1.65)
		-2.5	-2.6	-2.6	-2.5	-2.5
ln(Delay) for Income less than $100,000 per year		**-0.583**	**-0.191**	**-0.506**	**-0.500**	**-0.489**
		(0.410)	(0.445)	(0.407)	(0.405)	(0.395)
		-1.4	-0.4	-1.2	-1.2	-1.2
ln(Delay) for Income $100,000 or more per year		**-2.676**	**-2.542**	**-2.624**	**-2.592**	**-2.530**
		(0.895)	(0.914)	(0.878)	(0.916)	(0.887)
		-3.0	-2.8	-3.0	-2.8	-2.9
Time on Numbered Routes		**0.090**	**0.098**	**0.078**	**0.078**	**0.076**
		(0.031)	(0.032)	(0.030)	(0.031)	(0.030)
		2.9	3.1	2.6	2.5	2.5
Least Distance Path Dummy		**0.759**	**0.987**	**0.728**	**0.717**	**0.693**
		(0.249)	(0.245)	(0.245)	(0.260)	(0.257)
		3.0	4.0	3.0	2.8	2.7
Least Estimated Time Path Dummy		**0.377**	**0.881**	**0.382**	**0.375**	**0.354**
		(0.244)	(0.222)	(0.239)	(0.244)	(0.245)
		1.5	4.0	1.6	1.5	1.5
ln(Path Size)	**Estimate**	**0.730**		**0.617**	**0.610**	**0.594**
Based on Free-Flow Time	(Std. Error)	(0.122)		(0.119)	(0.135)	(0.134)
$\gamma = \infty$	T-Stat w/r/t 0	6.0		5.2	4.5	4.5
	T-Stat w/r/t 1	-2.2		-3.2	-2.9	-3.0
Gamma	**Estimate**				**0.87**	**2.84**
Nesting Parameter	(Std. Error)				(0.97)	(5.73)
	T-Stat w/r/t 0				0.9	0.5
	T-Stat w/r/t 1				-0.1	0.3
Number of Coefficients		**12**	**11**	**12**	**13**	**13**
Log-Likelihood at Estimates		**-393.1**	**-404.1**	**-390.6**	**-390.6**	**-390.6**
Rho-bar-squared		**0.221**	**0.201**	**0.225**	**0.223**	**0.223**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70.

Table 5-9. Cross-Nested Logit Estimation Results.

Finally, parameterizing the nesting coefficients, μ_m, does not seem to have led to much improvement in fit. The estimates of the γ parameters in μ_m have such broad standard errors that it is difficult to make any statistical conclusion about their true values. Models estimated with the specification of Equation 5-3 also failed to improve fit, as γ_1 and γ_2 were highly correlated. We adopt the original Bekhor specification of Equation 2-1 for forecasting in section 5.6.

5.4.3 Logit Kernel

Results of Logit Kernel estimations are shown in Table 5-10. Recall that Maximum Simulated Likelihood Estimation (MSLE) is used for these specifications because of the large number of Gaussian variates (one per link in each respondent's choice set). It is not computationally feasible to estimate these structures using numerical integration. Therefore, the number of draws used in the simulation and estimation process is shown at the top of each column. Walker (2001) advises that the number of draws must be sufficiently large so that parameter results are "stable" or "robust" as the number of draws increases. This will be discussed in more detail later.

The coefficient estimates presented in Table 5-10 have the same signs as those presented for the PSL models in Table 5-5, so the general interpretation is unchanged. However, notice that the Logit Kernel estimates do not appear to be the same multiple of the PSL coefficients, and that the significance (that is, the standard errors and t-statistics) of some coefficients have changed. For example, the coefficient on distance did not increase as much compared to the coefficient on free-flow time. Also, the distance coefficient was statistically different from zero in the PSL specification, while this assertion cannot be made for the LK specification. In contrast, the coefficient on the log of delay for households with annual incomes under \$100,000 per year has higher t-statistics under the LK specification.

Coefficient		Number of Draws				
		10,000	**4,096**	**10,000**	**24,649**	**100,000**
Distance	**Estimate**	**-0.179**	**-0.274**	**-0.323**	**-0.327**	**-0.296**
	(Std. Error)	(0.445)	(0.471)	(0.448)	(0.450)	(0.441)
	T-Statistic	-0.4	-0.6	-0.7	-0.7	-0.7
Free-Flow Time		**-4.00**	**-3.66**	**-3.44**	**-3.34**	**-3.43**
		(0.55)	(0.87)	(1.47)	(2.01)	(1.71)
		-7.2	-4.2	-2.3	-1.7	-2.0
Mass. Pike Dummy		**-5.96**	**-5.65**	**-5.40**	**-5.19**	**-5.41**
		(3.43)	(3.52)	(3.82)	(4.34)	(4.06)
		-1.7	-1.6	-1.4	-1.2	-1.3
Tobin Bridge Dummy		**6.72**	**8.58**	**7.82**	**8.02**	**8.39**
		(4.49)	(5.36)	(5.91)	(6.54)	(6.21)
		1.5	1.6	1.3	1.2	1.4
Sumner Tunnel Dummy		**4.46**	**5.25**	**4.96**	**4.57**	**4.81**
		(7.73)	(7.83)	(7.77)	(7.75)	(7.57)
		0.6	0.7	0.6	0.6	0.7
ln(Delay) for No Income Reported		**-13.20**	**-13.50**	**-12.40**	**-12.5**	**-14.2**
		(6.54)	(7.07)	(7.68)	(9.63)	(9.22)
		-2.0	-1.9	-1.6	-1.3	-1.5
ln(Delay) for Income less than $100,000 per year		**-2.57**	**-3.06**	**-2.78**	**-2.82**	**-2.86**
		(1.68)	(1.72)	(1.89)	(2.25)	(2.04)
		-1.5	-1.8	-1.5	-1.3	-1.4
ln(Delay) for Income $100,000 or more per year		**-10.00**	**-10.80**	**-10.10**	**-10.00**	**-10.20**
		(5.55)	(5.84)	(6.50)	(7.45)	(6.91)
		-1.8	-1.9	-1.6	-1.3	-1.5
Time on Numbered Routes		**0.378**	**0.367**	**0.338**	**0.335**	**0.345**
		(0.127)	(0.142)	(0.165)	(0.200)	(0.182)
		3.0	2.1	2.1	1.7	1.9
Least Distance Path Dummy		**2.52**	**1.82**	**1.73**	**1.67**	**1.73**
		(0.80)	(0.85)	(0.96)	(1.13)	(1.03)
		3.2	2.1	1.8	1.5	1.7
Least Estimated Time Path Dummy		**2.10**	**1.32**	**1.26**	**1.26**	**1.19**
		(0.71)	(0.74)	(0.77)	(0.86)	(0.79)
		2.9	1.8	1.6	1.5	1.5
ln(Path Size)	**Estimate**		**1.13**	**0.99**	**1.07**	**1.03**
Based on Free-Flow Time	(Std. Error)		(0.51)	(0.49)	(0.54)	(0.52)
$\gamma = \infty$	T-Stat w/r/t 0		2.2	2.0	2.0	2.0
	T-Stat w/r/t 1		0.3	-0.0	0.1	0.1
Sigma	**Estimate**	**2.17**	**2.12**	**1.99**	**1.97**	**1.97**
Gaussian Covariance Parameter	(Std. Error)	(0.37)	(0.58)	(0.88)	(1.18)	(1.00)
Based on Free-Flow Time	T-Stat w/r/t 0	5.83	3.7	2.3	1.7	2.0
Number of Coefficients		**12**	**13**	**13**	**13**	**13**
Log-Likelihood at Estimates		**-390.0**	**-382.4**	**-382.4**	**-382.3**	**-382.0**
Rho-bar-squared		**0.226**	**0.239**	**0.239**	**0.239**	**0.240**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70. Number of draws are chosen to be perfect squares; e.g., 4,096 = 64^2 and 24,649 = 157^2.

Table 5-10. Logit Kernel Estimation Results.

In the LK specifications with Path Size, the Path Size coefficient is statistically different from zero and not significantly (or in some cases, much numerically) different from one. The estimate

In the LK specifications with Path Size, the Path Size coefficient is statistically different from zero and not significantly (or in some cases, much numerically) different from one. The estimate of the Gaussian covariance parameter, σ, is also statistically different from zero. This suggests that the Gaussian covariances, which are proportional to path overlap free-flow times, capture a different effect than the Path Size. Also notice that the LK specifications with Path Size have a better fit than the CNL specifications with Path Size presented in Table 5-9.

Notice that for the LK specification with Path Size, the standard errors increased when the number of draws increases from about 4 thousand to about 25 thousand draws. For the LK estimation runs with Path Size, the optimizer reported a condition that suggests a local maximum. We conclude that these results are not stable or reliable.

Also notice that the parameter estimates for the Logit Kernel models have a different scale or magnitude than the PSL and other Logit Family models presented earlier. Recall that for the Logit Kernel model, the error term in the route utilities consists of a Gaussian term (which may be correlated with the corresponding terms for other routes) and an independent Gumbel term. In the earlier Logit Family models presented, route utilities have only the independent Gumbel term. Therefore, the Gumbel term of the Logit Family models has a greater variance than the Gumbel term of the Logit Kernel models. Also recall that the variance of a Gumbel variable is inversely proportional to its scale parameter, μ. Further recall that Logit Family and Logit Kernel models are normalized to a Gumbel with scale parameter $\mu = 1$, or alternatively, the coefficient estimates presented are really estimates of $\mu\beta$ jointly. Therefore, we expect the Logit Kernel parameters to have a larger scale than the Logit Family coefficients. (See Appendix E for more on scaling.)

Table 5-11 presents the Logit Kernel estimates of 5-10 scaled so that the coefficient on free-flow time has the same magnitude as Model 4 of Table 5-5. The scaled coefficients on distance, log of delay for high-income travelers and the label dummies have a much smaller magnitude than the corresponding PSL coefficients. However, notice that the Path Size coefficients have a greater magnitude and are quite close to one.

Coefficient		**Number of Draws**				
		10,000	**4,096**	**10,000**	**24,649**	**100,000**
Distance	**Estimate**	**-0.023**	**-0.038**	**-0.048**	**-0.050**	**-0.044**
	(Std. Error)	(0.057)	(0.066)	(0.067)	(0.069)	(0.066)
	T-Statistic	-0.4	-0.6	-0.7	-0.7	-0.7
Free-Flow Time		**-0.513**	**-0.513**	**-0.513**	**-0.513**	**-0.513**
		(0.071)	(0.112)	(0.219)	(0.309)	(0.256)
		-7.2	-4.2	-2.3	-1.7	-2.0
Mass. Pike Dummy		**-0.76**	**-0.79**	**-0.81**	**-0.80**	**-0.81**
		(0.44)	(0.49)	(0.57)	(0.67)	(0.61)
		-1.7	-1.6	-1.4	-1.2	-1.3
Tobin Bridge Dummy		**0.86**	**1.20**	**1.17**	**1.23**	**1.26**
		(0.58)	(0.75)	(0.88)	(1.00)	(0.93)
		1.5	1.6	1.3	1.2	1.4
Sumner Tunnel Dummy		**0.57**	**0.74**	**0.74**	**0.70**	**0.72**
		(0.99)	(1.10)	(1.16)	(1.19)	(1.13)
		0.6	0.7	0.6	0.6	0.7
ln(Delay) for No Income Reported		**-1.69**	**-1.89**	**-1.85**	**-1.92**	**-2.12**
		(0.84)	(0.99)	(1.15)	(1.48)	(1.38)
		-2.0	-1.9	-1.6	-1.3	-1.5
ln(Delay) for Income less than $100,000 per year		**-0.330**	**-0.429**	**-0.415**	**-0.433**	**-0.428**
		(0.216)	(0.241)	(0.282)	(0.346)	(0.305)
		-1.5	-1.8	-1.5	-1.3	-1.4
ln(Delay) for Income $100,000 or more per year		**-1.283**	**-1.514**	**-1.507**	**-1.537**	**-1.526**
		(0.712)	(0.819)	(0.970)	(1.145)	(1.034)
		-1.8	-1.9	-1.6	-1.3	-1.5
Time on Numbered Routes		**0.0485**	**0.0515**	**0.0504**	**0.0515**	**0.0516**
		(0.0163)	(0.0201)	(0.0246)	(0.0307)	(0.0272)
		3.0	2.1	2.1	1.7	1.9
Least Distance Path Dummy		**0.323**	**0.255**	**0.258**	**0.257**	**0.259**
		(0.103)	(0.119)	(0.143)	(0.174)	(0.154)
		3.2	2.1	1.8	1.5	1.7
Least Estimated Time Path Dummy		**0.269**	**0.185**	**0.188**	**0.194**	**0.178**
		(0.092)	(0.104)	(0.115)	(0.132)	(0.119)
		2.9	1.8	1.6	1.5	1.5
ln(Path Size)	**Estimate**		**1.13**	**0.99**	**1.07**	**1.03**
Based on Free-Flow Time	(Std. Error)		(0.51)	(0.49)	(0.54)	(0.52)
$\gamma = \infty$	T-Stat w/r/t 0		2.2	2.0	2.0	2.0
	T-Stat w/r/t 1		0.3	-0.0	0.1	0.1
Sigma	**Estimate**	**0.278**	**0.297**	**0.297**	**0.303**	**0.295**
Gaussian Covariance Parameter	(Std. Error)	(0.048)	(0.081)	(0.131)	(0.181)	(0.149)
Based on Free-Flow Time	T-Stat w/r/t 0	5.83	3.7	2.3	1.7	2.0
Number of Coefficients		**12**	**13**	**13**	**13**	**13**
Log-Likelihood at Estimates		**-390.0**	**-382.4**	**-382.4**	**-382.3**	**-382.0**
Rho-bar-squared		**0.226**	**0.239**	**0.239**	**0.239**	**0.240**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70. Number of draws are chosen to be perfect squares; e.g., 4,096 = 64^2 and 24,649 = 157^2.

Table 5-11. Scaled Logit Kernel Estimation Results.

We also examined random coefficient specifications for the route choice model, which are presented in Table 5-12. Note that the magnitude and significance of the utility parameters generally tracks that found in other models. The model in the second column may be thought of as the MNL model (see the first column, which is reproduced from Table 5-7) with a random free-flow time coefficient. In this model, the free-flow time coefficient maintains its strong significance (a t-statistic of –6.0, compared to –6.6 for MNL), and the random free-flow time coefficient is marginally significant. There is little change in the goodness of fit between this model and MNL.

In the fourth column, we present estimation results from a model with both a random free-flow time coefficient and a path size term. Notice that the parameter values are not different from the base PSL model (which is shown in the third column). However, the standard errors and t-statistics of this model are different. Most importantly, the random coefficient was estimated to have a large standard error as it was strongly correlated with free-flow time. Note that the path size term is derived from free-flow time, but path size was not estimated to have a large correlation with the random free-flow time coefficient. Also note that the log-likelihood of the PSL + RC model is not different from that of the PSL model without random coefficients.

We obtained similar results for models with the LK covariance structure for path overlap and the parameter sigma – the likelihood did not improve when we added random coefficients. (See the continuation of Table 5-12.) One difficulty of estimating the LK + PS + RC specification is that the likelihood is no longer globally concave. When we estimated this structure starting with zero coefficient values, we obtained a local maximum result where the magnitude of the random free-flow time coefficient was about one-fifth that of the free-flow time coefficient. However, these estimates resulted in a log-likelihood less than that of the LK + PS results.

To verify the estimation results and to obtain the true global maximum, we used a line search procedure where the random free-flow time coefficient was constrained to various values between zero and the unconstrained (local maximum) estimate. The final column of Table 5-12 shows the result for a very small value of the random coefficient. Model likelihood decreased as the value of the random coefficient increased, so we conclude the global optimum is the LK + PS specification without a random coefficient.

Coefficient		MNL	MNL+RC	PSL	PSL+RC
Distance	**Estimate**	**-0.253**	**-0.238**	**-0.212**	**-0.212**
	(Std. Error)	(0.106)	(0.093)	(0.100)	(0.092)
	T-Statistic	-2.4	-2.6	-2.1	-2.3
Free-Flow Time		**-0.601**	**-0.694**	**-0.513**	**-0.513**
		(0.091)	(0.116)	(0.082)	(1.640)
		-6.6	-6.0	-6.3	-0.3
Std. Dev. of Random Free-Flow Time Coeff.			**0.337**		**0.002**
			(0.190)		(23.600)
			1.8		0.0
Mass. Pike Dummy		**-0.64**	**-0.84**	**-0.49**	**-0.50**
		(0.68)	(0.75)	(0.60)	(0.70)
		-0.9	-1.1	-0.8	-0.7
Tobin Bridge Dummy		**2.90**	**2.76**	**2.75**	**2.75**
		(0.94)	(0.95)	(0.88)	(0.89)
		3.1	2.9	3.1	3.1
Sumner Tunnel Dummy		**2.18**	**2.03**	**1.92**	**1.92**
		(1.20)	(1.25)	(1.15)	(1.24)
		1.8	1.6	1.7	1.6
ln(Delay) for No Income Reported		**-5.25**	**-4.75**	**-4.45**	**-4.42**
		(1.93)	(2.13)	(1.77)	(1.74)
		-2.7	-2.2	-2.5	-2.5
ln(Delay) for Income less than $100,000 per year		**-0.328**	**-0.251**	**-0.583**	**-0.582**
		(0.445)	(0.478)	(0.410)	(0.412)
		-0.7	-0.5	-1.4	-1.4
ln(Delay) for Income $100,000 or more per year		**-2.70**	**-2.79**	**-2.68**	**-2.67**
		(0.95)	(1.59)	(0.90)	(1.46)
		-2.9	-1.8	-3.0	-1.8
Time on Numbered Routes		**0.110**	**0.101**	**0.090**	**0.099**
		(0.032)	(0.030)	(0.031)	(0.032)
		3.4	3.4	2.9	2.8
Least Distance Path Dummy		**0.994**	**1.03**	**0.759**	**0.759**
		(0.257)	(0.25)	(0.249)	(0.243)
		3.9	4.2	3.0	3.1
Least Estimated Time Path Dummy		**0.971**	**0.899**	**0.377**	**0.378**
		(0.227)	(0.239)	(0.244)	(0.223)
		4.3	3.8	1.5	1.7
ln(Path Size)	**Estimate**			**0.730**	**0.730**
Based on Free-Flow Time	(Std. Error)			(0.122)	(0.158)
$\gamma = \infty$	T-Stat w/r/t 0			6.0	4.6
	T-Stat w/r/t 1			-2.2	-1.7
Number of Coefficients		**11**	**12**	**12**	**13**
Log-Likelihood at Estimates		**-410.8**	**-409.7**	**-393.1**	**-393.1**
Rho-bar-squared		**0.188**	**0.189**	**0.221**	**0.219**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70. MNL+RC and PSL+RC models use 10,000 draws. LK+PS and LK+PS+RC models use 100,000 draws, and are scaled so the free-flow-time coefficient is the same as the PSL model. The std. dev. of random free-flow time coefficient in the LK+PS+RC model was calibrated using a line search process. Likelihood is maximized when this coefficient is near or at zero.

Table 5-12. Logit Kernel with Random Parameters Estimation Results.

Coefficient		LK+PS	LK+PS+RC
Distance	**Estimate**	**-0.044**	**-0.045**
	(Std. Error)	(0.066)	(0.066)
	T-Statistic	-0.7	-0.7
Free-Flow Time		**-0.513**	**-0.513**
		(0.256)	(0.332)
		-2.0	-1.6
Std. Dev. of Random Free-Flow Time Coeff.			**0.2×10^{-6}**
			(constrained)
Mass. Pike Dummy		**-0.81**	**-0.82**
		(0.61)	(0.69)
		-1.3	-1.2
Tobin Bridge Dummy		**1.26**	**1.26**
		(0.93)	(1.02)
		1.4	1.2
Sumner Tunnel Dummy		**0.72**	**0.70**
		(1.13)	(1.16)
		0.7	0.6
ln(Delay) for No Income Reported		**-2.12**	**-2.10**
		(1.38)	(1.66)
		-1.5	-1.3
ln(Delay) for Income less than $100,000 per year		**-0.428**	**-0.425**
		(0.305)	(0.352)
		-1.4	-1.2
ln(Delay) for Income $100,000 or more per year		**-1.53**	**-1.54**
		(1.03)	(1.16)
		-1.5	-1.3
Time on Numbered Routes		**0.0516**	**0.0518**
		(0.0272)	(0.0324)
		1.9	1.3
Least Distance Path Dummy		**0.259**	**0.258**
		(0.154)	(0.180)
		1.7	1.4
Least Estimated Time Path Dummy		**0.178**	**0.178**
		(0.119)	(0.128)
		1.5	1.4
ln(Path Size)	**Estimate**	**1.03**	**1.00**
Based on Free-Flow Time	(Std. Error)	(0.52)	(0.52)
$\gamma = \infty$	T-Stat w/r/t 0	2.0	1.9
	T-Stat w/r/t 1	0.1	0.0
Sigma	**Estimate**	**0.295**	**0.293**
Gaussian Covariance Parameter	(Std. Error)	(0.149)	(0.187)
Based on Free-Flow Time	T-Stat w/r/t 0	2.0	-1.6
Number of Coefficients		**13**	**13**
Log-Likelihood at Estimates		**-382.0**	**-382.0**
Rho-bar-squared		**0.240**	**0.240**

Table 5-12. Logit Kernel with Random Parameters Estimation Results (Continued).

These results suggest reviewing the conclusions of Han, Algers and Engleson (2001) in a more critical light. Recall that Han, Algers and Engleson present random coefficients models of stated preference route choices. They report that the addition of random coefficients significantly improves the fit of their models, though some coefficients will have counter-intuitive signs for a fraction of their distribution. Our results suggest that random coefficients add only minor explanatory power, and that better fit is achieved with the path size term or LK covariance structure. While our sample is homogeneous (MIT employees); the Han, Algers and Engelson study did not address the issue of path overlap. Clearly, further exploration is needed in this aspect of route choice.

5.4.4 IAP Logit

We examined IAP Logit specification using the binary logistic availability model proposed by Cascetta and Papola (1998). This specification may be simplified as follows:

Recall the IAP choice probabilities are

$$P_n(i) = \frac{\exp\left(\sum_{k=1}^{K_U} X_{ink}\beta_k + \beta_{IAP}\ln a_{in}\right)}{\sum_{j\in M}\exp\left(\sum_{k=1}^{K_U} X_{jnk}\beta_k + \beta_{IAP}\ln a_{jn}\right)},$$

where K_U is the number of utility parameters, and a_{in} takes a value between zero and one depending on the extent to which the decision-maker perceives alternative *i*. In the binary logistic availability model,

$$a_{in} = \frac{1}{1+\exp\left(-\sum_{k=1}^{K_{IAP}} Y_{ink}\gamma_k\right)},$$

where K_{IAP} is the number of variables relating to availability or perception. Solving for ln a_{in}, we obtain

$$\ln a_{in} = \ln\left[\frac{1}{1+\exp\left(-\sum_{k=1}^{K_{IAP}} Y_{ink}\gamma_k\right)}\right] = \ln(1) - \ln\left[1+\exp\left(-\sum_{k=1}^{K_{IAP}} Y_{ink}\gamma_k\right)\right] = -\ln\left[1+\exp\left(-\sum_{k=1}^{K_{IAP}} Y_{ink}\gamma_k\right)\right]$$

The IAP choice probabilities then become

$$P_n(i) = \frac{\exp\left(\sum_{k=1}^{K_U} X_{ink}\beta_k - \beta_{IAP}\ln\left[1+\exp\left(-\sum_{k=1}^{K_{IAP}} Y_{ink}\gamma_k\right)\right]\right)}{\sum_{j\in M}\exp\left(\sum_{k=1}^{K_U} X_{jnk}\beta_k - \beta_{IAP}\ln\left[1+\exp\left(-\sum_{k=1}^{K_{IAP}} Y_{jnk}\gamma_k\right)\right]\right)}.$$

This formulation may then be estimated using software such as GAUSS. However, we were unable to obtain any acceptable results for the binary logistic specification of the IAP term. In some cases, the parameter estimates converged to a (possibly locally) optimal likelihood, but the hessian was non-invertible. This suggests there may be some identification issues that still need to be examined with this specification. One possible issue to consider is whether the β_{IAP} parameter is separately estimable (that is, identified). Also, although the traveler characteristic of network knowledge appears to be identifiable as a *Y* variable – since it is added to path attributes and transformed by the log function – it may be possible that some more complex interaction prevents it from being identified if it is included in the IAP terms (ln *a*) for all routes.

Instead we turned to a different transformation of network knowledge and LOS attributes. The intuition we wanted to capture is simple: a route is more likely to be known if a person has more network knowledge or if the route is more prominent relative to others in the choice set. If we scale network knowledge and a route attribute so that both take values between zero and one, we can use a transformation inspired by Boolean algebra:

$\ln a = \ln [\ 1 - (1 - NK)(1 - LOS)\ ]$

where *NK* and *LOS* are the transformed variables. The reader can easily calculate the values of *a* when *NK* and *LOS* take their extreme values of zero and one. (Treating the value one as "true," this is the logical OR operation.)

Scaling the structural network knowledge model shown in Figure 5-5 (by dividing by the maximum network knowledge predicted for any individual) gives

$$NK_n = 0.0031 + 0.386\ MAPS_n + 0.463\ EXPERIENCE_n + 0.0031\ YEARS_n \qquad (5\text{-}4)$$

where $MAPS_n$ = 1 if individual *n* learned his or her habitual route from maps, and 0 otherwise;

$EXPERIENCE_n$ = 1 if individual *n* learned his or her habitual route through experience, and 0 otherwise; and

$YEARS_n$ is the number of years the respondent has lived in the Boston metropolitan area.

From earlier models, numbered route time seemed to be an important variable relating to prominence. The percentage of travel time on numbered routes is therefore a natural variable for prominence, and it has the characteristic that it is defined to range from zero to 100 percent. (We also examined a specification where percent of time on numbered routes is divided by the maximum percentage within each individuals' choice set, but this had a slightly lower likelihood.)

Table 5-13 presents the estimation results using the IAP term based on estimates of network knowledge from the MIMiC model of Figure 5-5. Estimates are shown for MNL, PSL, CNL and LK models. MNL and PSL models estimated without an IAP term are included for comparison. Note that the addition of the IAP term causes a marginal improvement in overall model fit. No IAP coefficient is significant at the 95 percent level of confidence, and the coefficient is most significant in the MNL+IAP and CNL+IAP models. When a Path Size term is added to any IAP model, the IAP term decreases in significance. This is not surprising as the motivation for both the PS and IAP terms is to account for travelers' perceptions.

Coefficient		MNL	PSL	MNL+IAP	PSL+IAP
Distance	**Estimate** (Std. Error) T-Statistic	**-0.253** (0.106) -2.4	**-0.212** (0.100) -2.1	**-0.240** (0.106) -2.3	**-0.200** (0.101) -2.0
Free-Flow Time		**-0.601** (0.091) -6.6	**-0.513** (0.082) -6.3	**-0.605** (0.092) -6.6	**-0.515** (0.083) -6.2
Mass. Pike Dummy		**-0.64** (0.68) -0.9	**-0.49** (0.60) -0.8	**-0.68** (0.69) -1.0	**-0.52** (0.61) -0.9
Tobin Bridge Dummy		**2.90** (0.94) 3.1	**2.75** (0.88) 3.1	**2.90** (0.94) 3.1	**2.76** (0.89) 3.1
Sumner Tunnel Dummy		**2.18** (1.20) 1.8	**1.92** (1.15) 1.7	**2.20** (1.20) 1.8	**1.95** (1.19) 1.7
ln(Delay) for No Income Reported		**-5.13** (1.98) -2.6	**-4.45** (1.77) -2.5	**-5.12** (2.01) -2.6	**-4.45** (1.79) -2.5
ln(Delay) for Income less than $100,000 per year		**-0.205** (0.441) -0.5	**-0.583** (0.410) -1.4	**-0.194** (0.448) -0.4	**-0.562** (0.416) -1.4
ln(Delay) for Income $100,000 or more per year		**-2.562** (0.936) -2.7	**-2.676** (0.895) -3.0	**-2.484** (0.931) -2.7	**-2.617** (0.891) -2.9
Time on Numbered Routes		**0.112** (0.032) 3.5	**0.090** (0.031) 2.9	**0.078** (0.038) 2.1	**0.059** (0.036) 1.7
Least Distance Path Dummy		**1.056** (0.251) 4.2	**0.759** (0.249) 3.0	**1.089** (0.253) 4.3	**0.795** (0.252) 3.2
Least Estimated Time Path Dummy		**0.971** (0.227) 4.3	**0.377** (0.244) 1.5	**0.963** (0.226) 4.3	**0.371** (0.244) 1.5
ln(Path Size) Based on Free-Flow Time $\gamma = \infty$	**Estimate** (Std. Error) T-Stat w/r/t 0 T-Stat w/r/t 1		**0.730** (0.122) 6.0 -2.2		**0.726** (0.123) 5.9 -2.2
ln a IAP term	**Estimate** (Std. Error) T-Stat w/r/t 0			**1.23** (0.69) 1.8	**1.10** (0.67) 1.7
Number of Coefficients		**11**	**12**	**12**	**13**
Log-Likelihood at Estimates		**-410.8**	**-393.1**	**-409.4**	**-391.6**
Rho-bar-squared		**0.188**	**0.221**	**0.190**	**0.222**

Table 5-13. IAP Logit with MIMiC Network Knowledge Estimation Results.

Coefficient		CNL+IAP	CNL+IAP +PSL	LK+IAP	LK+IAP +PSL
Distance	**Estimate**	**-0.242**	**-0.216**	**-0.021**	**-0.044**
	(Std. Error)	(0.103)	(0.099)	(0.053)	(0.067)
	T-Statistic	-2.4	-2.2	-0.4	-0.7
Free-Flow Time		**-0.557**	**-0.476**	**-0.601**	**-0.513**
		(0.086)	(0.079)	(0.051)	(0.327)
		-6.4	-6.0	-11.9	-1.6
Mass. Pike Dummy		**-0.57**	**-0.41**	**-0.86**	**-0.82**
		(0.66)	(0.60)	(0.49)	(0.70)
		-0.9	-0.7	-1.7	-1.2
Tobin Bridge Dummy		**2.79**	**2.76**	**1.13**	**1.19**
		(0.91)	(0.87)	(0.39)	(1.03)
		3.1	3.2	2.9	1.2
Sumner Tunnel Dummy		**2.09**	**1.95**	**0.81**	**0.72**
		(1.16)	(1.12)	(1.12)	(1.19)
		1.8	1.7	0.7	0.6
ln(Delay) for No Income Reported		**-4.80**	**-4.27**	**-2.36**	**-1.79**
		(1.88)	(1.68)	(0.96)	(1.48)
		-2.6	-2.5	-2.5	-1.2
ln(Delay) for Income less than $100,000 per year		**-0.182**	**-0.490**	**-0.379**	**-0.414**
		(0.452)	(0.413)	(0.249)	(0.352)
		-0.4	-1.2	-1.5	-1.2
ln(Delay) for Income $100,000 or more per year		**-2.471**	**-2.570**	**-1.531**	**-1.530**
		(0.909)	(0.875)	(0.781)	(1.160)
		-2.7	-2.9	-2.0	-1.3
Time on Numbered Routes		**0.066**	**0.051**	**0.0403**	**0.0319**
		(0.036)	(0.035)	(0.0207)	(0.0276)
		1.8	1.5	2.0	1.2
Least Distance Path Dummy		**1.021**	**0.764**	**0.382**	**0.274**
		(0.247)	(0.247)	(0.116)	(0.188)
		4.1	3.1	3.3	1.5
Least Estimated Time Path Dummy		**0.875**	**0.379**	**0.308**	**0.191**
		(0.222)	(0.239)	(0.102)	(0.133)
		3.9	1.6	3.0	1.4
ln(Path Size)	**Estimate**		**0.612**		**1.010**
Based on Free-Flow Time	(Std. Error)		(0.119)		(0.509)
$\gamma = \infty$	T-Stat w/r/t 0		5.1		2.0
	T-Stat w/r/t 1		-3.3		0.0
Sigma	**Estimate**			**0.325**	**0.301**
Gaussian Covariance Parameter	(Std. Error)			(0.039)	(0.188)
Based on Free-Flow Time	T-Stat w/r/t 0			8.3	1.6
ln a	**Estimate**	**1.13**	**1.00**	**0.63**	**0.62**
IAP term	(Std. Error)	(0.66)	(0.64)	(0.45)	(0.57)
	T-Stat w/r/t 0	1.7	1.6	1.4	1.1
Number of Coefficients		**12**	**13**	**13**	**14**
Log-Likelihood at Estimates		**-409.5**	**-389.3**	**-389.9**	**-380.7**
Rho-bar-squared		**0.202**	**0.226**	**0.225**	**0.240**

Notes: N = 159. Log-likelihood for all coefficients at zero is -519.70. 24,649 draws were used for Logit Kernel estimation. Logit Kernel coefficients are scaled to have the same free-flow-time coefficient as the corresponding MNL or PSL model.

Table 5-13. IAP Logit with MIMiC Network Knowledge Estimation Results (Continued).

Also note that the coefficient on time on numbered routes decreases in magnitude and significance when the IAP term is added. This is also to be expected, because the IAP term is derived from the time on numbered routes, so correlation between these two terms is to be expected.

Finally, note that network knowledge is a calculated quantity, and therefore subject to error. It is possible to estimate a similar IAP model specification directly by assuming a positive correlation between network knowledge and its explanatory variables, that is, years in Boston, use of maps and learning from experience. However, the technique presented below does not make use of the survey indicators, which are found in the measurement equations of the MIMiC model.

Network knowledge scaled to take values between zero and one can then be expressed as

$$\begin{aligned} NK_n &= \frac{\gamma_1 YEARS_n + \gamma_2 MAP_n + \gamma_3 EXPERIENCE_n}{\max_{n=1 \text{ to } N}\left(\gamma_1 YEARS_n + \gamma_2 MAP_n + \gamma_3 EXPERIENCE_n\right)} \\ &= \frac{\gamma_1 YEARS_n + \gamma_2 MAP_n + \gamma_3 EXPERIENCE_n}{\gamma_1 \max_n YEARS_n + \gamma_2 \max_n MAP_n + \gamma_3 \max_n EXPERIENCE_n} \\ &= \frac{\gamma_1 YEARS_n + \gamma_2 MAP_n + \gamma_3 EXPERIENCE_n}{\gamma_1 \max_n YEARS_n + \gamma_2 + \gamma_3} \end{aligned}$$

since MAP_n and $EXPERIENCE_n$ are dummy variables. Since we want the γ coefficients to be positive, we can re-parameterize them as $\gamma_k = \exp(\gamma'_k)$. Also note that we cannot identify all of the γ coefficients in the formulation above, since we could multiply each γ by a constant and still obtain the same value of *NK*. For estimation, we choose arbitrarily to set γ_1 equal to one (that is, γ'_1 is set to zero). However, for interpreting model results, it is more meaningful to scale the γ coefficients so that fitted Network Knowledge will take values between zero and one. (That is, calculating

$$\frac{\gamma_k}{\gamma_1 \max_n YEARS_n + \gamma_2 + \gamma_3} \tag{5-5}$$

instead of presenting γ_k.)

Estimation results for the above specification are shown in Table 5-14. Note that the utility coefficients and t-statistics are comparable to those of previous models. However, the β coefficient on the ln(a) term is considerably different from those of the combined MIMiC+IAP models shown in Table 5-13. Also, the coefficient on the IAP term – that is, on ln(a) – is significant in the models of Table 5-14, while it was not for the MIMiC+IAP ones. The γ' coefficient on learning from maps is very significant, while the coefficient on experience is not very significant. Both direct models show a substantial improvement in likelihood with respect to the corresponding MIMiC+IAP model of Table 5-13.

It is more meaningful to interpret the direct IAP Logit coefficients as the scaled gammas given by Equation 5-5, so they may be compared with the MIMiC coefficients. This comparison is shown in Table 5-15. Note that the direct IAP coefficients on tenure in the Boston region have a greater magnitude, while those on learning from experience are smaller. This result may be promising for evaluating future scenarios, as tenure in a region may be more easily forecast.

We now turn to the discussion of the suitability of the various model types for other prospective route choice applications.

5.5 Comparison of Route Choice Model Types

As discussed in Section 5.4, estimation results for the various model types are presented to examine the usefulness of the Path Size Logit model in relation to other available route choice models: MNL, C-Logit, CNL, LK, IAP or some combination. Table 5-16 presents a summary of the results of Sections 5.3 and 5.4 (Tables 5-5 though 5-14). Individual coefficient estimates and standard errors are not presented here; since all utility coefficients have reasonable signs, this table focuses on goodness-of-fit measures and the properties of the coefficients related to the

Coefficient	MNL+IAP	PSL+IAP
Distance **Estimate** (Std. Error) T-Statistic	**-0.254** (0.108) -2.4	**-0.220** (0.102) -2.2
Free-Flow Time	**-0.646** (0.094) -6.9	**-0.551** (0.086) -6.4
Mass. Pike Dummy	**-0.61** (0.68) -0.9	**-0.47** (0.60) -0.8
Tobin Bridge Dummy	**2.95** (0.94) 3.1	**2.80** (0.88) 3.2
Sumner Tunnel Dummy	**2.07** (1.20) 1.7	**1.87** (1.15) 1.6
ln(Delay) for No Income Reported	**-5.32** (1.99) -2.7	**-4.61** (1.78) -2.6
ln(Delay) for Income less than $100,000 per year	**-0.360** (0.455) -0.8	**-0.673** (0.420) -1.6
ln(Delay) for Income $100,000 or more per year	**-2.558** (0.938) -2.7	**-2.660** (0.900) -3.0
Time on Numbered Routes	**0.175** (0.058) 3.0	**0.146** (0.056) 2.6
Least Distance Path Dummy	**1.049** (0.252) 4.2	**0.747** (0.252) 3.0
Least Estimated Time Path Dummy	**1.013** (0.228) 4.5	**0.441** (0.245) 1.8
ln(Path Size) **Estimate** Based on Free-Flow Time (Std. Error) $\gamma = \infty$ T-Stat w/r/t 0 T-Stat w/r/t 1		**0.703** (0.123) 5.7 -2.4

Table 5-14. IAP Logit with Estimation of Network Knowledge Results.

Coefficient		MNL+IAP	PSL+IAP
ln a IAP term	**Estimate** (Std. Error) T-Stat w/r/t 0	**4.80** (1.34) 3.6	**4.12** (1.30) 3.2
γ_1' **IAP Coefficient on Years in Boston**		0*	0*
γ_2' **IAP Coefficient on Learning from Maps**		**2.90** (0.37) 7.8	**2.89** (0.42) 6.9
γ_3' **IAP Coefficient on Learning from Experience**		**1.31** (1.92) 0.7	**1.36** (2.15) 0.6
Number of Coefficients		**14**	**15**
Log-Likelihood at Estimates		**-403.5**	**-387.3**
Rho-bar-squared		**0.197**	**0.226**

Note: * indicates a constrained coefficient.

Table 5-14. IAP Logit with Estimation of Network Knowledge Results (Continued).

IAP Coefficient on	Estimates from		
	MIMiC	Direct MNL+IAP	Direct PSL+IAP
Years in Boston	0.0031	0.0127	0.0127
Learning from Maps	0.386	0.231	0.228
Learning from Experience	0.463	0.047	0.050

Notes: Coefficients presented are scaled as shown by Equation 5-5.

Table 5-15. Comparison of MIMiC and Direct IAP Logit Network Knowledge Coefficient Estimates.

error structure introduced by overlapping paths. For example, all of the models with a Path Size term yielded coefficient estimates that were significant and consistent with theory.

The Commonality Factor of C-Logit did not perform as expected. Without Path Size, the coefficient on the Commonality Factor had a t-statistic of about 1.5. When the Path Size term was included, the Commonality Factor coefficient lost significance and changed to an unexpected sign. The coefficients related to both the Cross-Nested Logit and Logit Kernel structures were significant and had values consistent with theory. As discussed in Section 5.4.2, Cross-Nested Logit produced better fit than MNL, but not as good as PSL. CNL with a Path Size term performed better than either PSL or CNL alone. Logit Kernel alone had the best fit of any of the single model types (MNL, PSL, C-Logit and CNL). Logit Kernel with a Path Size term had the best fit overall, but required more draws than "pure" Logit Kernel.

As was done with the analysis of Path Generation algorithms, we also consider the computational effort required for each of these model types. These are presented in Table 5-18 below. Only the time of the estimation routine is presented; that is, the time required for data preparation is omitted from the table. Data preparation time is roughly constant for all model types. All use a common file containing levels-of-service variables and traveler characteristics. The terms unique to each model – Path Sizes, Commonality Factors, alphas for CNL, and covariance terms for LK – can all be calculated quickly in TransCAD given the link-path incidence matrix from the path generation step.

Path-Size Logit and C-Logit apply additive correction terms to MNL utilities, and can therefore be estimated with standard MNL software, such as ALOGIT. This software is well known and quite optimized. An IAP Logit model with a fully-specified IAP term (as opposed to one in which coefficients are to be estimated) can also be estimated with MNL software.

IAP Logit with a binary logistic IAP term (and parameters to be estimated) can be estimated simultaneously in GAUSS. Since this formulation requires repeated calls to the MNL probability function (once for the IAP terms and once for the overall route choice probabilities), it runs about as fast as MNL software. Similarly, simultaneous estimation of the network knowledge and IAP model in GAUSS has relatively quick computational time. In both cases, longer running times may indicate that the model is not identified or a suitable convergence cannot be reached.

Property	MNL	PSL	C-Logit	C-Logit with PS	CNL	CNL with PS	LK	LK with PS
Results shown on Table	5-7	5-5	5-8	5-8	5-9	5-9	5-10 5-11	5-10 5-11
Path Size coefficient is consistent with theory and significant?		Y		Y		Y		Y
Commonality Factor coefficient is consistent with theory?			Y	N				
Commonality Factor coefficient is significant?			?	N				
CNL coefficients are consistent with theory and significant?					Y	Y		
LK coefficient (s) is consistent with theory and significant?							Y	Y
Log-Likelihood at Estimates	-410.8	-393.1	-409.6	-392.5	-404.1	-390.6	-390.0	-382.0
Rho-Bar-Squared	0.188	0.221	0.189	0.220	0.201	0.225	0.226	0.240

Notes: Blank cells indicate a property is not applicable to a particular model type.
Table contents refer to the best specification for each model type. The same utility specification (X variables) is used throughout.
N = 159 and log-likelihood at zero is –519.7 for all models.
MNL = Multinomial Logit; PS(L) = Path-Size (Logit); CNL = Cross-Nested Logit; and LK = Logit Kernel.

Table 5-16. Summary of Alternative Model Specifications.

Description of Estimation Routine	Workstation	Time to Estimate
MNL including PSL and C-Logit (ALOGIT)	A	about 5 min
Cross-Nested Logit (GAUSS)	B	4.5 h
Parameterized Cross-Nested Logit (GAUSS)	B	6 h
Logit Kernel (Bolduc Code) – 4,096 draws	B	1.5 h
Logit Kernel – 10,000 draws	B	6.75 h
Logit Kernel – 24,649 draws	B	9.25 h
Logit Kernel – 100,000 draws	B	2.25 d

Notes: Workstation A is a 60 MHz Pentium with 16 MB RAM running Windows 95.
Workstation B is a 933 MHz Pentium III with 256 MB RAM running Windows NT 4.0.

Table 5-17. Comparison of Computational Time for Estimation.

Cross-Nested Logit and Logit Kernel are much more computationally involved. The Logit Kernel running time is an increasing function of the number of draws. In the Boston Case Study, the running times for CNL and LK are about the same. Network size, topology, and the maximum number of alternative paths considered will of course affect the computational times of the CNL and LK software.

It is worth noting at this point that the Logit Kernel software has been highly optimized by Bolduc (2001). For example, only the memory required for the number of links considered by each observation is used by the LK code.

In contrast, the CNL code allocates memory for the same number of links for each observation – this number must then be the maximum number of links in any individual's choice set – about 850 for the Boston Case Study. Therefore observations using fewer links have matrices (such as alphas) filled with many zeros. The CNL code could be further optimized for memory use; the impact of this refinement on running time is not immediately clear.

The results shown suggest that the combined Path Size and Logit Kernel model is promising. The CNL specification also appears to be worth further examination by researchers. In particular, specifications of alpha and mu should be explored to verify whether empirical results with fit comparable to the PSL results presented (see Table 5-7) can be obtained.

Both Path Size with Logit Kernel and Path Size with CNL have better fit than PSL alone. Both model types require significant computational time and expertise to estimate. If these resources are available, we recommend LK or CNL with a Path Size term. If not, plain PSL should be estimated, as it represents about the same computational effort as MNL and yields much better fit.

Path Size is useful because of its easy implementation, and with appropriate calibration, it fits observed travel choices well. Exponential Path Size with gamma = infinity seems to work best for the Boston data set. This is not surprising to the extent that large gammas were expected to be favored to avoid the problem of Path Sizes producing a "penalty" to utility, and therefore choice probabilities in counter-intuitive directions from those of MNL. However, the result that infinity is the best-fitting value of gamma *is* surprising, as this suggests the basis underlying the Path Size calculation is deterministic. We expect Exponential PS with a large value of gamma (e.g., 99) to work best, or maybe a Generalized Path Size formulation using exp($\bullet$) or some other relation as the $G(\bullet)$ function.

Path Size and the Gaussian overlap error structure (refer to Table 5-10) seem to capture different aspects of perception: The Gaussian structure reflects path overlapping and correlation of travel conditions, while Path Size reflects traveler perceptions of the "reasonableness" or "applicability" of a path to solving the problem of getting from O to D. Quick paths and distinct paths have sizes of one. For a path to have a small size, it must overlap with other paths that are quicker – this makes it suboptimal and therefore less relevant to solving the problem.

C-Logit does not produce useful results for the Boston data set. Since PSL has better empirical fit (see Table 5-8) and a stronger theoretical basis, it dominates C-Logit.

The IAP Logit model may benefit from further exploration. The model presented with a single interaction as the IAP term shows some improvement in fit, although its parameter is not strong statistically. IAP Logit terms with a binary logistic formulation have not produced satisfactory models to date. This result may indicate an underlying problem with identification.

Finally, it is worth noting that terms involving network knowledge and prominence appear to belong in route choice models and help their explanatory power. The IAP term, time on numbered routes, label dummies, facility dummies (to some extent) fall into this category.

5.6 Comparison of Predictions from Different Model Types

Applying a PSL, CNL or LK model is not much more difficult than estimating one. For example, the link-path incidence matrix that will be needed during the network loading step can be the one produced in the path generation step before model estimation. Only if the network has changed appreciably between the estimation and application settings – primarily through the addition or deletion of links – would a new link-path matrix need to be generated. Note that when simulation is used as a path generation procedure, the choice set contained in the link-path matrix already embodies a wide variety of travel times. Therefore, if traffic conditions change between the estimation and forecast year, for example, a new link-path matrix would only be needed if forecast travel times are substantially beyond those used in the distribution for simulated path generation.

Applying PSL, CNL or LK for route choice or traffic assignment would then involve the following steps: (1) calculating new path skims from the existing link-path matrix and new link attributes, if necessary; (2) calculating the route choice probabilities; and (3) loading the predicted trips to the appropriate links. The first and third steps are essentially database manipulations. The second step corresponds to an iteration of the MLE estimation routine. Therefore, we do not expect applying the PSL model to take substantially longer than estimation, or to require “calibration” (minor adjustment of model parameters to produce estimated traffic counts and other aggregate measures that more consistent with network observations; as the initial parameter estimates are based on a limited number of travelers) that is more intensive than for other types of route choice models.

With appropriate programming, PSL may be used as the network loading routine of a user equilibrium assignment. Current computational times for CNL and LK application may make them prohibitive for equilibrium loading. However, it is worth noting that results by Prashker and Bekhor (1998) suggest that as levels of congestion increase, or as more iterations of traffic assignment are run and assigned volumes approach that of user equilibrium, the individual network loading procedure (e.g., shortest paths, STOCH, or PSL) has less of an effect on the final volumes. Of course, more distributed network loaders may lead to faster convergence of the UE assignment. (That is, STOCH and PSL may be preferred over all-or-nothing assignment.)

We examine an application of the route choice models developed above by using sample enumeration. The model types we will consider are MNL, CNL and LK with Path Size and IAP variants. Note we do not consider any deterministic (for example, shortest path) models, as the choice set generation analysis highlighted the difficulty in using a single objective or label to identify paths travelers are likely to choose. We will test sensitivity to “policy” scenarios – or different demographic assumptions – by comparing the change in the predicted choice probabilities for selected respondents. The scenarios we consider are as follows:

(1) Reduced signage. The motivation for this scenario is the common complaint that roadway signage is not as plentiful as in other states, and that desired information is often missing from existing signage. This scenario assumes a constant signage budget, and that the Massachusetts Highway Department chooses a subset of “significant” routes – for example, freeways and expressways, and their continuation – for targeted sign improvements. Other routes would be dropped from the state numbering system.

We assume the that routes remaining in the highway system under this scenario are all Interstates and expressways; U.S. Highways 1, 20 and 44; and Massachusetts State Routes 1A, 2, 3, 9, 28, 99, 107 and 203.

Testing this scenario involves changing the "numbered route" attribute of affected links. The "time on numbered routes" coefficient could be adjusted to reflect the better quality of signage on remaining numbered routes, but we do not consider this option.

(2) New Harbor and River Crossings. In this scenario, the aging Tobin Bridge is removed and replaced with a new bridge further up the Mystic River, as has been proposed. We assume that this new crossing will be designated as "US 1." Because of its proximity to the Mystic River Parkway, we assume this facility will not have tolls. The Ted Williams Tunnel (I-90 Extension) has also been added. No facility constant is assumed for the New Mystic River Bridge or the Ted Williams Tunnel, as these would have to be estimated from new data or calibrated from observed volumes. These facilities were assumed to have free-flow speeds and congestion levels similar to the facilities they replace (in the case of the new bridge) or parallel (in the case of the Ted Williams Tunnel). These changes are shown graphically in Figure 5-6. Because of the changes to the physical network, this scenario requires new path generation.

(3) In-vehicle guidance. This scenario represents an extreme in information availability, which we simulate by modifying driver socio-economic characteristics. To reflect the depth of knowledge available from an in-vehicle route guidance system, we will set the "years in Boston" variable to its maximum in the data set – 57 years. We will also set the "learned from maps" variable to one, as we assume that this navigation device will have the same content as static maps. Note that under the assumptions of the MNL or STOCH model, full information would imply a deterministic or AON assignment.

Because scenario 2 involves a particular section of the network, we evaluate all scenarios with a subset of travelers originating in the North Shore. These travelers would likely have their

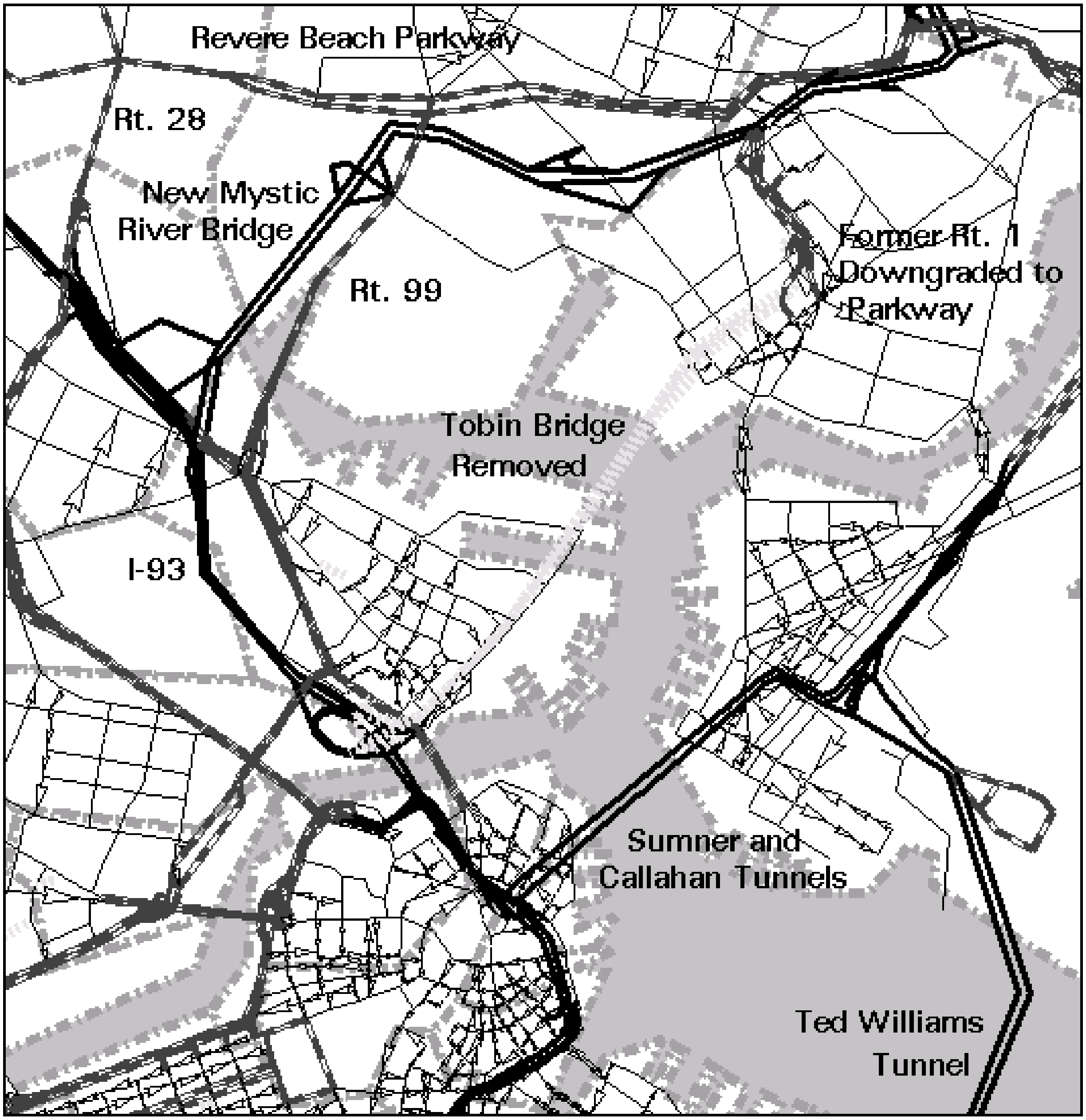


Figure 5-6. Network Under New Mystic River Bridge Scenario.

commuting patterns altered by the removal of the Tobin Bridge. Characteristics of the selected travelers are shown in Table 5-18. Travelers were selected to represent a wide variety of commuting distances, income and network knowledge.

The changes in individual choice probabilities under the reduced signage scenario are presented in Table 5-19. Note that in this table, predictions are grouped by heavy lines into three model types: MNL; "modified" logit, including PSL, CNL and the combination of the two; and Logit Kernel. Note that the two subjects from Nahant are not affected by this scenario, as their commuting routes would remain in the numbered highway network.

The respondent from Winthrop is less likely to use the habitual route under this scenario. The habitual route uses a non-expressway section of Rt. 145 to Rt. 1A – one of the longest sections dropped from the state highway system in this person's choice set. Therefore, the forecasts predict that this traveler will now be more willing to use other roads to access Rt. 1A and the Sumner Tunnel. Also note that MNL predicts the greatest change to the habitual route choice probability. All models predict an increased probability of using the least-distance route, with the LK variants predicting the greatest increase. The least-distance route for this respondent uses primarily expressway links, and is relatively unaffected by the removal of highway signs. The cross-nested and LK models predict an increased probability of using the minimum-time route (for this respondent, this route minimizes both free-flow and estimated time, and maximizes utility), while the remaining models predict a decrease. This result is unexpected and may highlight the different representation of the correlation among routes in CNL and LK in contrast to MNL and PSL. This sign difference may also be caused by this route having an intermediate change in numbered route time, which produces different diversions under the different model specifications. Also note that PSL predicts the greatest change for the minimum time route, a decrease of about 11 percent. This sign difference in forecasts is also present for the routes using the Tobin Bridge. All models predict a reduction in Sumner Tunnel use, again with PSL predicting the greatest change in probability.

Respondent Number and Origin	Income Group	Years in Boston	Learned from Maps?	Learned by Experience?	Scaled Network Knowledge
1257 – Gloucester	3	44	N	Y	0.60
298 – Winthrop	2	37	N	Y	0.58
1339 – Newburyport	4	57	Y	N	0.57
318 – Nahant	6	19	N	Y	0.53
919 – Nahant	2	42	N	N	0.13

Notes: Income group 2 corresponds to $25,000 to $49,999 annual household income; group 3 to $50,000 to $74,999; group 4 to $75,000 to $99,999; and group 6 to $150,000 or more.

Fitted network knowledge is based on MIMiC model coefficients and scaled to take a value between zero and one. Please see equation 5-5. Respondents are sorted by scaled network knowledge.

Table 5-18. Characteristics of Travelers Considered in Applications.

Respondent #298 - Winthrop						
Absolute Change in Predicted Choice Probability and [Base Choice Probability] for:	**MNL**	**PSL**	**CNL**	**CNL+ PS**	**LK**	**LK + PS**
Habitual Route Pauline St., Sumner Tunnel, Central Artery, Storrow Drive, Longfellow Bridge, Portland	-1.5% [5.9%]	-0.5% [4.9%]	-0.4% [4.1%]	-0.5% [4.1%]	-0.7% [4.2%]	-0.9% [4.3%]
Minimum Distance Route Winthrop St., Revere Beach Parkway, Fellsway, Gore St., Cardinal Medieros, Portland	+0.8% [2.2%]	+2.4% [5.2%]	+1.6% [2.0%]	+2.4% [4.6%]	+6.0% [10.9%]	+5.5% [13.4%]
Minimum Free-Flow Time Route = Minimum Est. Time Route Winthrop St., Sumner Tunnel, Central Artery, Storrow Drive, Longfellow Bridge, Broadway, Ames, Main, Portland	-1.1% [11.0%]	-11.1% [39.7%]	+3.9% [24.7%]	+2.9% [37.9%]	+3.3% [34.8%]	+2.4% [41.5%]
Maximum Utility Route	-1.1% [11.0%]	-11.1% [39.7%]	+3.9% [24.7%]	+2.9% [37.9%]	+3.3% [34.8%]	+2.4% [41.5%]
Routes Using Tobin Bridge	+0.6% [1.6%]	+1.3% [2.8%]	-0.2% [1.5%]	-0.5% [2.8%]	-0.7% [3.5%]	-1.3% [5.3%]
Routes Using Sumner Tunnel	-2.1% [93.9%]	-4.6% [89.8%]	-1.2% [94.2%]	-1.7% [90.3%]	-2.3% [78.7%]	-2.4% [76.5%]

Table 5-19. Change in Choice Probabilities for Reduced Signage Scenario.

Respondent #1257 – Gloucester

Absolute Change in Predicted Choice Probability and [Base Choice Probability] for:	MNL	PSL	CNL	CNL+ PS	LK	LK + PS
Habitual Route Rt. 128, Rt. 1, Revere Beach Pkwy, Fellsway, Rutherford Ave., Gilmore Br., Mem. Dr.	NC [1.2%]	NC [1.3%]	+1.1% [1.4%]	+1.1% [1.5%]	+1.6% [2.9%]	+1.8% [2.5%]
Minimum Distance Route Rt. 128, Rt. 1, Rt. 99, Revere Beach Pkwy., Fellsway, McGrath & O'Brien Hwy., Mem. Dr.	NC [5.1%]	NC [2.2%]	-4.2% [5.0%]	-2.0% [2.6%]	-3.4% [4.1%]	-3.0% [3.5%]
Minimum Free-Flow Time Route Rt. 128, I-93, Storrow Dr. Exit, Longfellow Br., Mem. Dr.	+0.1% [6.9%]	+0.3% [26.1%]	-4.1% [6.0%]	-11.2% [19.4%]	-7.2% [7.5%]	-24.9% [28.1%]
Minimum Est. Time Route Rt. 128, I-93, Storrow Dr. Exit, Science Park Br., Mem. Dr.	NC [5.4%]	NC [4.3%]	-0.8% [10.8%]	NC [5.6%]	-7.2% [21.1%]	-2.5% [12.2%]
Maximum Utility Route	+0.1% [6.9%]	+0.3% [26.1%]	-0.8% [10.8%]	-11.2% [19.4%]	-7.2% [21.1%]	-18.3% [28.1%]
Routes Using Tobin Bridge	NC [8.2%]	+0.1% [11.3%]	+0.3% [9.7%]	+1.0% [14.2%]	+0.7% [6.9%]	-1.3% [10.0%]
No Routes Use Sumner Tunnel						

Table 5-19. Change in Choice Probabilities for Reduced Signage Scenario (Continued).

Respondent #1339 – Newburyport						
Absolute Change in Predicted Choice Probability and [Base Choice Probability] for:	**MNL**	**PSL**	**CNL**	**CNL+ PS**	**LK**	**LK + PS**
Habitual Route Rt. 113, I-95, Rt. 1, Tobin Br., Gilmore Br., Mem. Dr., Binney St., Galileo Way	NC [1.5%]	NC [1.1%]	NC [1.6%]	NC [1.4%]	NC [0.9%]	NC [0.9%]
Minimum Distance Routes* Rt. 113, I-95, Rt. 1 [minor deviations], Rt. 99, Revere Beach Pkwy., Fellsway, McGrath & O'Brien Hwy., Gore St., Fulkerson, Galileo Way	+0.3% [32.1%]	+0.3% [35.9%]	+0.3% [31.4%]	+0.3% [34.5%]	NC [14.6%]	+0.2% [17.7%]
Minimum Free-Flow Time Route = Minimum Est. Time Route Rt. 113, I-95, Rt. 128, I-93, Storrow Dr. Exit, Longfellow Br., Broadway, Galileo Way	NC [5.2%]	+0.1% [12.2%]	NC [4.5%]	+0.1% [9.1%]	NC [16.2%]	+0.3% [20.2%]
Maximum Utility Route	+0.2% [19.3%]	+0.3% [31.4%]	+0.2% [18.7%]	+0.3% [29.1%]	NC [16.2%]	+0.3% [20.2%]
Routes Using Tobin Bridge	+0.3% [31.8%]	+0.2% [26.5%]	+0.3% [31.5%]	+0.3% [28.1%]	+0.2% [17.3%]	+0.2% [21.6%]
No Routes Use Sumner Tunnel						

Notes: * indicates the sum of choice probabilities for two or more routes that "tie" to optimize a label. No physical path description is shown for the Maximum Utility Route, as this route may change based on model specification or scenario.

NC = No change (that is, less than 0.1 percent), MNL = Multinomial Logit, PS(L) = Path-Size (Logit), CNL = Cross-Nested Logit, and LK = Logit Kernel.

Table 5-19. Change in Choice Probabilities for Reduced Signage Scenario (Continued).

For the individual from Gloucester, the cross-nested and LK models predict the greatest changes in choice probabilities – for some routes, MNL and PSL do not predict a noticeable change. This respondent also exhibits conflicting signs between the cross-nested and LK versus the MNL and PSL predictions for the minimum free-flow-time route and maximum utility route probabilities. The CNL and LK models that include a Path Size term produce large changes in the choice probability for the minimum free-flow-time route. The minimum free-flow-time route is among those with the least decrease in sign-posted roadways, so we expect it to retain or attract market share. Note that the minimum free-flow-time route is also the maximum utility route for four of the six model specifications. Also, routes using the Sumner Tunnel are not in this individual's choice set, and therefore the tunnel is unused in both the base and scenario case.

None of the six models predict a change in the Newburyport respondent's habitual route. Minor increases in the choice probabilities of label routes are predicted, but it's likely that these changes are within the "noise" of the models. Note that the label paths from this respondent's choice set all lost roughly the same amount of signed roadway, so changes in choice probabilities should be minimal.

The results of the reduced signage scenario do not suggest any dramatic conclusions; in fact, the few sign changes across model types are troubling. However, this scenario does illustrate that in general, the inclusion of a path-size term can have a large impact on some predictions. Also, the results suggest that the predictions from the LK model variants tend to be different from the other model types.

Table 5-20 shows the predictions for the New Mystic River Bridge scenario. In this scenario, major construction results in a significant change to the physical network. This scenario requires new path generation, and in general, the label paths in the base case do not correspond to a path in the test case. For this reason, we present only the predicted change in choice probabilities for two major corridors crossing the Harbor and Mystic River: the Sumner and Ted Williams

Respondent #298 - Winthrop

Abs. Change in Predicted Choice Probability and [Base Choice Probability] for:	MNL	PSL	CNL	CNL+PS	LK	LK + PS
Routes Using Tobin Bridge or New Mystic River Bridge	+27.5% [1.6%]	+33.6% [2.8%]	+33.4% [1.5%]	+36.8% [2.8%]	+45.4% [3.5%]	+41.3% [5.3%]
Routes Using Sumner Tunnel or Ted Williams Tunnel	-22.9% [93.9%]	-26.2% [89.8%]	-29.1% [94.2%]	-29.8% [90.3%]	-27.5% [78.7%]	-24.8% [76.5%]

Respondent #318 – Nahant

Abs. Change in Predicted Choice Probability and [Base Choice Probability] for:	MNL	PSL	CNL	CNL+PS	LK	LK + PS
Routes Using Tobin Bridge or New Mystic River Bridge	+41.4% [54.8%]	+47.6% [51.8%]	+63.6% [36.1%]	+65.1% [34.3%]	+78.1% [21.4%]	+74.3% [24.9%]
Routes Using Sumner Tunnel or Ted Williams Tunnel	-13.3% [13.5%]	-15.3% [15.9%]	-28.8% [29.1%]	-31.3% [31.9%]	-37.7% [38.1%]	-37.6% [38.4%]

Respondent #919 – Nahant

Abs. Change in Predicted Choice Probability and [Base Choice Probability] for:	MNL	PSL	CNL	CNL+PS	LK	LK + PS
Routes Using Tobin Bridge or New Mystic River Bridge	+71.8% [27.6%]	+72.0% [26.8%]	+70.6% [28.8%]	+70.1% [28.7%]	+80.9% [18.1%]	+76.6% [21.9%]
Routes Using Sumner Tunnel or Ted Williams Tunnel	-26.7% [27.3%]	-29.0% [30.3%]	-24.6% [25.2%]	-27.7% [28.9%]	-33.7% [34.8%]	-33.5% [35.1%]

Table 5-20. Change in Choice Probabilities for New Mystic River Bridge Scenario.

<table>
<tr><th colspan="7">Respondent #1257 – Gloucester</th></tr>
<tr><th>Abs. Change in Predicted Choice Probability and [Base Choice Probability] for:</th><th>MNL</th><th>PSL</th><th>CNL</th><th>CNL+PS</th><th>LK</th><th>LK + PS</th></tr>
<tr><td><b>Routes Using Tobin Bridge or New Mystic River Bridge</b></td><td>+80.1%
[8.2%]</td><td>+76.0%
[11.3%]</td><td>+78.7%
[9.7%]</td><td>+72.7%
[14.2%]</td><td>+58.2%
[6.9%]</td><td>+55.5%
[10.0%]</td></tr>
<tr><td colspan="7"><b>No Routes Use Sumner Tunnel or Ted Williams Tunnel</b></td></tr>
<tr><th colspan="7">Respondent #1339 – Newburyport</th></tr>
<tr><th>Abs. Change in Predicted Choice Probability and [Base Choice Probability] for:</th><th>MNL</th><th>PSL</th><th>CNL</th><th>CNL+PS</th><th>LK</th><th>LK + PS</th></tr>
<tr><td><b>Routes Using Tobin Bridge or New Mystic River Bridge</b></td><td>+67.1%
[31.8%]</td><td>+70.3%
[26.5%]</td><td>+66.5%
[31.5%]</td><td>+67.0%
[28.1%]</td><td>+74.3%
[17.3%]</td><td>+67.2%
[21.6%]</td></tr>
<tr><td colspan="7"><b>No Routes Use Sumner Tunnel or Ted Williams Tunnel</b></td></tr>
</table>

Notes: NC = No change (that is, less than 0.1 percent), MNL = Multinomial Logit, PS(L) = Path-Size (Logit), CNL = Cross-Nested Logit, and LK = Logit Kernel.

Because of the significant change to network topology in this scenario, it no longer makes sense to talk about change to a habitual route. Specifically, if the respondent reported using the Tobin Bridge habitually, the habitual route is no longer feasible. Similarly, new physical routes may minimize labels under this scenario.

Table 5-20. Change in Choice Probabilities for New Mystic River Bridge Scenario (Continued).

Tunnels near Logan Airport, and the Mystic River Bridge designated U.S. 1 (currently the Tobin Bridge, to become the new bridge in this scenario).

The huge change in this network results in a substantial change in the levels-of-service variables for the alternative routes available to all respondents. This in turn results in large changes in choice probabilities – changes of 70 percent for U.S. 1 are not uncommon. Were the purpose of this exercise to examine demands for this new bridge, an equilibrium assignment would be required. However, keep in mind that this is a sample enumeration exercise designed to illustrate the sensitivities of the various model specifications. Travel times and speeds for the new facilities in this network were arbitrarily assumed. (Alternatively, since this is a hypothetical exercise, we may say that the number of lanes and other design parameters of the new bridge have not been fixed, and therefore we are not interested in capacity restraint in this context.)

The LK models predict the largest changes in the probability the Winthrop resident uses U.S. 1, about 40 to 45 percent. The modified logit models predict about a 35 percent increase, while MNL predicts a more modest 27.5 percent change. Most all of this increase comes from diversion from the Sumner Tunnel.

For respondent #318 of Nahant, the LK and cross-nested models predict larger changes than the MNL and PSL specifications. Addition of a Path Size term to an MNL or CNL model results in a slightly increased probability for U.S. 1. However, the LK + PS model predicts a lesser shift to U.S. 1 than LK alone. Diversion from the Sumner and Ted corridor to U.S. 1 is less substantial for this respondent. Individual #318 has a high household income, and thus is less sensitive to paying the larger tunnel tolls. This individual is less likely to use slower expressway and arterial routes (such as the Fellsway or Rt. 99) in favor of the freeway leading to the New Mystic River Bridge.

In contrast, individual #929 from Nahant has a more modest income (and less network knowledge), and is predicted to have an even greater diversion to the New Mystic River Bridge.

All of the six models predict at least a 70 percent increase for U.S. 1, with the LK variants predicting the greatest increase. The LK models also predict the greatest diversion from the Sumner/Ted corridor.

Neither the Gloucester or Newburyport resident use the Sumner or Ted Williams tunnels; these facilities are just too far away from U.S. 1, which the respondents are likely to use inside Rt. 128. Increased choice probabilities for U.S. 1 indicate a diversion from expressway and arterial routes.

The MNL model predicts about an 80 percent increase in U.S. 1's probability for the Gloucester resident. The PSL and CNL models predict almost as large increases, and the CNL + PSL model predicts an approximately 70 percent increase. The LK models predict a more modest diversion to U.S. 1. The LK model predicts the greatest increase for the Newburyport resident's use of U.S. 1, though all models show a roughly 70 percent change in choice probability.

The New Mystic River Crossing scenario shows more clearly the existence of distinct "bands" among the MNL, modified logit and LK predictions. That is, each of these broad model classes tends to produce numerical predictions that do not vary much from those of other models of the same class, but are different from those of other model classes. In general, the LK predictions are usually different from both the MNL and modified logit estimates. The modified logit estimates may be close in value to the MNL predictions, or they may take an intermediate value between the MNL and LK probabilities.

This scenario also illustrates an interesting property of the path-size term: Models with path size tend to amplify or accentuate smaller changes, while attenuating larger changes towards 50 percent. Clearly, this property will have considerable implications for forecasting and the conclusions drawn from such forecasts.

Next, we turn to Table 5-21, which explores the choice probabilities the five respondents would have with in-vehicle navigation devices. Only IAP models will be sensitive to this scenario, so

Respondent #298 - Winthrop					
Absolute Change in Predicted Choice Probability and [Base Choice Probability] for:	Direct IAP + PS	MIMiC + IAP +...		MIMiC + IAP +...	
		CNL	CNL+PS	LK	LK + PS
Habitual Route	-0.6% [5.9%]	-0.1% [4.0%]	-0.1% [4.0%]	-0.2% [4.3%]	-0.2% [4.3%]
Minimum Distance Route	+0.4% [4.4%]	NC [2.0%]	+0.1% [4.6%]	+0.1% [10.4%]	+0.1% [13.5%]
Minimum Free-Flow Time Route = Minimum Est. Time Route	-2.2% [43.9%]	-0.5% [24.7%]	-0.6% [37.7%]	-0.6% [35.1%]	-0.7% [41.7%]
Maximum Utility Route	-2.2% [43.9%]	-0.5% [24.7%]	-0.6% [37.7%]	-0.6% [35.1%]	-0.7% [41.7%]
Routes Using Tobin Bridge	NC [2.8%]	NC [1.4%]	NC [2.6%]	NC [3.6%]	NC [5.1%]
Routes Using Sumner Tunnel	-0.6% [90.9%]	-0.1% [94.4%]	-0.1% [90.5%]	-0.2% [79.9%]	-0.2% [76.6%]

Table 5-21. Change in Choice Probabilities under In-Vehicle Route Guidance.

Respondent #318 – Nahant					
Absolute Change in Predicted Choice Probability and [Base Choice Probability] for:	**Direct**	**MIMiC + IAP +...**		**MIMiC + IAP +...**	
	IAP + PS	**CNL**	**CNL+PS**	**LK**	**LK + PS**
Habitual Route Rt. 1A, Revere Beach Pkwy., Fellsway, McGrath and O'Brien Hwy., Mem. Dr., Broadway	-0.2% [6.8%]	-0.2% [10.9%]	-0.1% [8.1%]	-0.1% [11.6%]	-0.1% [10.1%]
Minimum Distance Route Rt. 1A, Revere Beach Pkwy., Fellsway, McGrath and O'Brien Hwy., Mem. Dr., Broadway	-0.2% [6.8%]	-0.2% [10.9%]	-0.1% [8.1%]	-0.1% [11.6%]	-0.1% [10.1%]
Minimum Free-Flow Time Route = **Minimum Est. Time Route** Rt. 1A, Revere Beach Pkwy., Fellsway, I-93, Storrow Dr. Exit, Longfellow Br.	-2.3% [21.2%]	-0.4% [10.6%]	-0.5% [15.2%]	-0.4% [14.9%]	-0.5% [17.6%]
Maximum Utility Route	-2.3% [21.2%]	-0.8% [15.2%]	+0.2% [18.6%]	+0.2% [20.6%]	+0.2% [22.0%]
Routes Using Tobin Bridge	+1.8% [46.9%]	+0.6% [36.2%]	+0.3% [34.7%]	+0.1% [22.3%]	+0.1% [24.3%]
Routes Using Sumner Tunnel	-1.0% [18.5%]	-0.6% [29.3%]	-0.3% [31.9%]	-0.4% [40.4%]	-0.4% [37.5%]

Table 5-21. Change in Choice Probabilities under In-Vehicle Route Guidance (Continued).

Respondent #919 – Nahant					
Absolute Change in Predicted Choice Probability	**Direct**	**MIMiC + IAP +...**		**MIMiC + IAP +...**	
and [Base Choice Probability] for:	**IAP + PS**	**CNL**	**CNL+PS**	**LK**	**LK + PS**
Habitual Route Rt. 1A, Revere Beach Pkwy., Fellsway, McGrath and O'Brien Hwy., Gore St., Fulkerson, Galileo Way	+0.8% [8.9%]	+0.2% [10.3%]	+0.2% [9.4%]	+0.2% [11.0%]	+0.2% [10.6%]
Minimum Distance Route Rt. 1A, Revere Beach Pkwy., Fellsway, McGrath and O'Brien Hwy., Mem. Dr., Broadway, Galileo Way	+0.8% [8.9%]	+0.2% [10.3%]	+0.2% [9.4%]	+0.2% [11.0%]	+0.2% [10.6%]
Minimum Free-Flow Time = **Minimum Est. Time Route** Rt. 1A, Revere Beach Pkwy., Fellsway, I-93, Storrow Dr. Exit, Longfellow Br., Broadway, Galileo Way	-0.9% [13.2%]	-0.3% [8.8%]	-0.4% [13.5%]	-0.3% [10.6%]	-0.4% [13.5%]
Maximum Utility Route	+0.6% [17.7%]	-0.6% [13.8%]	+0.1% [16.2%]	+0.1% [18.9%]	+0.1% [19.7%]
Routes Using Tobin Bridge	-0.2% [28.6%]	+0.2% [29.0%]	+0.1% [29.0%]	NC [19.8%]	NC [22.0%]
Routes Using Sumner Tunnel	-0.8% [33.8%]	-0.6% [25.3%]	-0.4% [28.9%]	-0.4% [36.6%]	-0.4% [34.2%]

Table 5-21. Change in Choice Probabilities under In-Vehicle Route Guidance (Continued).

Respondent #1257 – Gloucester					
Absolute Change in Predicted Choice Probability and [Base Choice Probability] for:	**Direct**	**MIMiC + IAP +…**		**MIMiC + IAP +…**	
	IAP + PS	**CNL**	**CNL+PS**	**LK**	**LK + PS**
Habitual Route	+0.1% [1.0%]	NC [1.4%]	NC [1.5%]	+0.1% [2.5%]	+0.1% [2.7%]
Minimum Distance Route	NC [2.5%]	NC [4.3%]	NC [2.4%]	-0.1% [3.3%]	-0.1% [2.8%]
Minimum Free-Flow Time Route	-1.5% [29.3%]	-0.1% [5.9%]	-0.4% [19.0%]	-0.1% [6.8%]	-0.6% [26.7%]
Minimum Est. Time Route	NC [4.4%]	-0.1% [11.1%]	-0.1% [5.8%]	NC [19.1%]	-0.1% [12.4%]
Maximum Utility Route	-1.5% [29.3%]	-0.1% [11.1%]	-0.4% [19.0%]	NC [19.1%]	-0.6% [26.7%]
Routes Using Tobin Bridge	+0.6% [9.8%]	+0.1% [9.5%]	+0.1% [14.0%]	NC [8.3%]	NC [9.7%]
No Routes Use Sumner Tunnel					

Table 5-21. Change in Choice Probabilities under In-Vehicle Route Guidance (Continued).

Respondent #1339 – Newburyport					
Absolute Change in Predicted Choice Probability	Direct	MIMiC + IAP +...		MIMiC + IAP +...	
and [Base Choice Probability] for:	IAP + PS	CNL	CNL+PS	LK	LK + PS
Habitual Route	+0.1% [1.0%]	NC [1.7%]	NC [1.4%]	NC [1.6%]	NC [0.9%]
Minimum Distance Routes*	+0.6% [36.9%]	-0.1% [30.5%]	-0.1% [33.7%]	NC [9.7%]	-0.1% [17.0%]
Minimum Free-Flow Time Route = Minimum Est. Time Route	-0.6% [9.6%]	NC [4.6%]	-0.1% [9.1%]	NC [10.1%]	-0.1% [19.9%]
Maximum Utility Route	+0.4% [32.1%]	NC [18.3%]	-0.1% [28.5%]	NC [10.1%]	-0.1% [19.9%]
Routes Using Tobin Bridge	-1.2% [31.2%]	-0.1% [31.1%]	-0.1% [27.8%]	NC [18.7%]	-0.1% [20.4%]
No Routes Use Sumner Tunnel					

Notes: * indicates the sum of choice probabilities for two or more routes that "tie" to optimize a label.
No physical path description is shown for the Maximum Utility Route, as this route may change based on model specification or scenario.
NC = No change (that is, less than 0.1 percent), PS(L) = Path-Size (Logit), CNL = Cross-Nested Logit, LK = Logit Kernel, and IAP = Implicit Availability/Perception.
Please see Table 5-20 for descriptions of physical routes corresponding to each label for respondents #298, #1257 and #1339.

Table 5-21. Change in Choice Probabilities under In-Vehicle Route Guidance (Continued).

we examine a different set of models: the direct IAP model with a path size term (from Table 5-14), and the CNL and LK models with IAP terms based on the MIMiC model of network knowledge (see Table 5-13). We do not choose to adjust the scale of the utility parameters of non-IAP models arbitrarily, as we have no good empirical basis for choosing the magnitude of the adjustment.

Recall that the IAP model estimation suggested that highway signage was most closely correlated with prominence; that is, the IAP term is a function of the time spent on numbered roadways. Therefore, we expect in-vehicle navigation to cause drivers to have an increased propensity for using routes with lower proportions of sign-posted roadways.

For the Winthrop resident, the minimum-distance route is the only one to show an increase, and the greatest increase is predicted by the direct IAP + PS model. This result parallels that of the reduced signage scenario, where the minimum-distance route was consistently predicted to experience an increase in choice probability. Of the label routes, the minimum-distance path has the least percentage of time spent on numbered roadways. Also, the minimum-distance route is fairly circuitous, so we would not expect many drivers to know it. Recall that anchor point theory suggests that inexperienced travelers are most likely to know routes that are close to a straight line constructed between their origin and destination.

The habitual route has the greatest percentage of numbered roadways (despite this driver being fairly knowledgeable), so, as expected, navigation leads to diversion from this route. With guidance, the Winthrop resident is also led away from the Sumner Tunnel, which had a large initial choice probability.

The direct IAP and PSL model predicts about a two percentage decrease in the choice probability for respondent #318 (of Nahant) using the minimum time route. More modest changes are predicted for the habitual and minimum distance route. This result may reflect that the minimum time route uses more prominent facilities (particularly I-93) for a greater duration than the habitual or minimum distance route. It may be surprising that guidance leads this driver to favor the Tobin Bridge, as it is a prominent facility. However, note that the base share for the Tobin is about 20 to 45 percent, and that to access the Tobin, this resident must use less prominent roads.

The changes for the label paths predicted by all four combined MIMiC + IAP models are similar for Nahant respondent #919, while the direct IAP model predicts similar but larger changes. As with respondent #318, the minimum time route for respondent #919 uses the more prominent I-93. The decrease in the maximum utility route predicted by the MIMiC + IAP + CNL model – when others predict an increase – results from the fact that in this model, a different route – with a higher percentage of numbered roadways – is the utility-maximizing one. Accordingly, we expect this route's probability to decrease. A similar reason may apply for the different signs for the Tobin Bridge probability predicted by the MIMiC + IAP + CNL models. In any case, the small magnitude of these changes does not provide cause for concern. All five models predict a marginal decrease in Sumner Tunnel use for this respondent.

With guidance, the driver from Gloucester is predicted to use the minimum free-flow time path less often. This path is very simple, as it is largely Interstates (I-95 and I-93). Note that of the sequential models, those with LK and/or path size specifications predict a greater difference in probability. A guided driver is also marginally more likely to use the Gloucester respondent's habitual route, which involves a lower proportion of sign-posted roads. Guidance also least to a greater probability for the Tobin Bridge, which initially had a choice probability of about 10 to 15 percent.

A guided driver coming from Newburyport is predicted to use the Tobin Bridge less than the survey respondent, which corroborates with the Tobin's prominence for this O-D pair. The minimum-time route is one of the most prominent, so the navigation device diverts travel from it to other routes (including the label routes shown). As with the Gloucester origin, the Newburyport forecasts show a greater choice probability change from the MIMiC + IAP models that include Path Size terms – the changes predicted by the sequential models without path size are barely discernable. Further, the direct IAP model predictions again have greater magnitudes than those of the sequential models.

Finally, it is worth mentioning the caveat that the MIMiC + IAP models of Table 5-21 do not predict larger than a one percent change in choice probabilities for any route or traveler. The direct IAP model predicts no more than a three percent change. This small magnitude of change is likely within model error, and is not unexpected. Recall that only the directly-estimated IAP Logit models of Table 5-14 had t-statistics of two or greater; unlike those presented in Table 5-13. The limited explanatory power of the IAP term in the MIMiC + IAP models is illustrated by the limited sensitivity to these scenarios. While the results from the Boston data set do not provide much sensitivity to information-related scenarios, we believe this discussion illustrates the usefulness of the IAP technique. For example, the diversions occur as expected. We expect that better data sets – with questions better adapted to examining and explaining network knowledge – and more opportunistic model specifications – for example a simultaneous IAP + PS model that includes survey indicators of network knowledge in a MIMiC specification – would lead to more significant IAP models and more sensitive policy applications. (This improvement is one of the extensions we discuss in the following chapter.)

5.7 Conclusions

The path set generation results of section 5.1 show that this step is critical to the route choice modeling process, as the minimum-distance and minimum-time paths do not appear to be those selected by a majority of travelers. The simulation technique of making random draws of link impedances appears to be a promising method of generating alternative routes with a wide spatial distribution at reasonable computational costs.

Section 5.2 confirms the hypothesis that network knowledge affects route choice, and establishes that attitudinal surveys are useful in developing models of network knowledge that can be incorporated in route choice models.

Many assertions may be made from the estimation results of sections 5.3 and 5.4. First, the path size term improved the fit of all route choice model types investigated, and deserves serious consideration for inclusion in future route choice models. The C-Logit models with ad hoc commonality factor specifications did not offer good model fit. However, we assert that the path-size term is a promising, behavioral-based way to specify a commonality factor. Any analysts interested in estimating a C-Logit model should consider a path size term such as that shown in Equation 3-3.

The Logit Kernel Probit model with Path Size showed the best fit of all classes of models considered. While adding a random coefficient improved the fit of the MNL model, once a path size term or LK Probit covariance structure is added to path utilities, the random coefficient was no longer significant. Since path size and the LK covariance structure have clear interpretations, these terms are to be preferred to random coefficients.

The CNL + PS model showed some improvement in fit over PSL. However, it is not clear whether this improved fit justifies the additional complexity required to estimate a CNL model. CNL + PS specifications merit further consideration, and we expect that their performance will be dependent upon the choice data and travel context.

The IAP Logit models examined offered a marginal improvement to model fit, and the ability to consider policy variables related to information and awareness. Data issues appear to have limited the effectiveness of the IAP models. Overall, LK + PS + IAP models seem to offer the most promising means of explaining route choice in the context of heterogeneous driver network knowledge. Clearly, we need data sets with more observations, and from different cities and trip purposes to estimate composite LK + PS + IAP models. Such data sets will also offer further evidence supporting the above assertions, or suggest the ways in which they are refuted.

Finally, the model application exercises of section 5.6 provide further insight into the differences among the route choice model types. The New Mystic River Bridge scenario showed that MNL, CNL and LK forecasts can be substantially different. Also, the addition of a path size term seems to amplify small changes in response to policy scenarios, and to attenuate larger responses. The in-vehicle navigation and reduced signage scenarios illustrate that traveler response to information may not always follow simple intuition, especially in a network context where travelers interact with other drivers. It is therefore important to develop better means to assess network knowledge and incorporate it in the route choice process.

6 Conclusions

This chapter provides a concluding overview of the implications of this research. First, section 6.1 discusses the contributions of this effort. Succinctly, this thesis has developed and presented a framework by which drivers' network knowledge may be estimated and included in route choice models. Section 6.2 describes how this framework is being applied to other route choice data sets and how experience from the Boston case study may improve future data collection efforts. The section also offers some suggestions for further refinement of route choice modeling.

6.1 Contributions

The main contribution of this thesis is the development of a methodology for analyzing route choices in the context of limited information, with the goal of developing models that are sensitive to knowledge and ATIS policies, activities, initiatives. This methodology consists of three components: (1) assessing driver attitudes and characteristics relating to wayfinding and knowledge, (2) assessing the ability to generate "reasonable" or "realistic" paths, and (3) modeling route choice in a way that reflects driver perceptions of the network, particularly path overlapping.

The first component (assessing network knowledge) involves a change in data collection practices, and we show typical questions one might ask in Appendix B. We present a methodology to relate network knowledge to demographic and attitudinal data, through the MIMiC model of sections 3.3.1 and 5.2.

The second component involves the methodology to compare alternative path generation algorithms by coding observed paths, trying different path generation algorithms, and having a metric with which to compare the algorithms (that is, coverage, described in section 3.2.1). This methodology allows development of a path generation procedure to proceed with objective evaluations, rather than relying on the analyst's intuition alone. (In some studies, it is the analysts' intuition that generates a small number of alternative routes, which is even more arbitrary than selecting path generation algorithms intuitively!) By applying this methodology, we identified simulation (see section 3.2.5) as an easily-implemented and quick technique for generating alternative routes.

The surprisingly low coverage results presented in section 5.1 show that path generation is a harder problem than many practitioners might suppose. We have presented evidence that the traditional method of assigning traffic to the minimum distance or time path does not accurately reflect how drivers select routes.

The final component of this framework involves developing and evaluating a choice modeling structure that best empirically describes travelers' perceptions, preferences and observed behavior. The development of the Path Size Logit model is presented in sections 2.1.10 and 3.1.2. The development of the Logit Kernel Probit model is presented in section 3.3.2. Estimation and application results are presented in sections 5.3 through 5.6. Simply estimating the PSL, CNL or LK models for a large network is an accomplishment that has not been done before. Recall that for the Logit Kernel model, the size of the Gaussian covariance structure is determined by the number of links in the choice set, the maximum of which is 856 for the Boston data. Therefore, the estimation results presented in section 5.4.3 involve calculating an 856-dimensional integral of the Gaussian CDF by simulation.

Being able to compare MNL, C-Logit, PSL, CNL and LK models (with possible IAP variants) is a further increment to this contribution. Note that because of the unique LK error structure, scaling is required to compare its coefficient estimates with those from the logit family of models. Two alternative scaling procedures are presented: one based on constraining the coefficient of a route attribute (as is done in section 5.4.3); the other based on assuming comparable variances of the errors in LK and logit models (see Appendix E). We also presented some evidence regarding the relationship between the number of draws and the stability of simulated LK parameter and probability estimates in Section 3.3.2.

In estimating and comparing these model structures, we have formalized their specifications for route choice and proposed general parameterizations to improve model fit. For example, section 3.1.2 presents several Path Size specifications and their resulting choice probabilities for sample networks. It presents a general specification from which different variants may be created and tested. We show the conditions under which the PSL model may make counter-intuitive forecasts, and offer guidance in selecting parameterizations to avoid such a problem. We show which variant fits the Boston data best, offer an explanation, and make recommendations for future PSL models. We also offer an intuitive explanation of Path Size theory, for example, its relation to choice from a menu and the notion of a link size contribution.

For the CNL model, we present a more flexible way of parameterizing the nesting coefficients from network topology in section 5.4.2. The first application of Logit Kernel to route choice is presented in section 3.3.2.

The value of being able to compare these different model structures is to empirically determine which best reflects drivers' perceptions of overlapping routes. This thesis has presented other methods to incorporate drivers' perceptions of the transportation network: Simple logit results show the usefulness of including non-traditional variables in a route choice model, such as numbered route time, and label and facility dummies. We also adapt the IAP Logit methodology of Cascetta and Papola (1998) to allow network knowledge to be used as an explanatory variable in route choice models. An alternative specification of the IAP term (presented in section 5.4.4) was developed to overcome the difficulties encountered with a binary logistic IAP term.

While the IAP coefficient estimates were not the strongest, we believe this approach offers promise for exploring how drivers' knowledge and the information they receive affect route choices. There are several possible explanations for the marginally significant results we present in Section 5.4.4. First, we are concerned with estimating a habitual trip (that is, home to work) in the static context. ATIS proponents assert that the most benefits (that is, changed travel choices) from static information would occur on unfamiliar trips. For familiar trips, an ATIS offering real-time travel information would have more benefit to drivers than one offering only historical travel times or network geography. Also, Stern and Leiser (1988) showed evidence that drivers stop learning new routes after a few years tenure in an urban area. The respondents in our route choice data set have a much wider distribution of tenure in Boston. Given the continued interest in ATIS and the participation of geographers in developing better means of assessing network knowledge, continued investigation of awareness of alternative routes appears to be justified.

Of course, IAP Logit is not the only way to model awareness of routes. Section 2.1.14 presented probabilistic choice set generation, which was not considered in this thesis. The major obstacle to be overcome is the large number of possible choice sets – for example, a universal choice set of 51 routes has about 2.3 x 10^{15} (that is, 2.3 quadrillion) possible subsets. Clearly some ingenious but realistic way of reducing this number of choice sets must be devised.

In section 5.6, we demonstrate how the models estimated in earlier sections of Chapter 5 can be used to test both traditional (that is, new facilities) and non-traditional (for example, information-related) policy scenarios. The framework presented in this thesis can also be used for considering network knowledge in other longer-range choices, as shown in Figure 2-4.

6.2 Products of This Thesis: Future Work

The value of this thesis' contribution can be established by its usefulness and relevance to other applications. This section considers such applications. Section 6.2.1 describes how the path generation methodology of Chapter 3 is being used for a data set from Lexington, Kentucky.

Section 6.2.2 discusses efforts to collect more meaningful data about network knowledge and attitudes toward wayfinding. Finally, sections 6.2.3 through 6.2.5 discuss other refinements to improve the behavioral realism of route choice modeling.

6.5.1 Lexington, KY, Case Study

Battelle (1997) reports on an ITS demonstration project to test the usefulness of GPS receivers for recording passenger travel data. About 100 households in Lexington, Kentucky, agreed to install a GPS receiver on the roof of their family car, and to record travel information in a vehicle-mounted terminal for a week. The participating households also kept a traditional paper travel diary for one day of the study. This rich data set continues to be examined by researchers. For example, Zhou and Golledge (2000) examine the activity data to find temporal and spatial patterns used by travelers. Jan, Horowitz and Peng (2000) explore drivers' variation in route choice for repeated trips between the same OD pair. Those authors make an interesting observations – few drivers in Lexington chose the shortest path. The authors are unable to offer an explanation, but the result is consistent with the observation that only about a third of the drivers in the Boston case study chose the least travel-time path (see Table 5-1). We propose three possible causes: (1) The drivers may be wishing to minimize some other objective function than time alone. (2) The drivers' perceptions of travel times may not match those in the network used by Jan, Horowitz and Peng. (3) The drivers may not be aware of all the links in the network, and thus are constrained to use links that may be suboptimal.

We are using the methodology presented in this thesis to examine the Lexington travel data. There are two motivations for this research effort: (1) to establish that the unexpected results from Boston drivers are reliable, and that travelers in many cities may choose routes other than the ones they would choose if they had better information; and (2) to compare estimates of PSL model parameters across cities of different sizes, network representations and demographic characteristics.

We constructed a network representation for Lexington from the Census TIGER files, using data provided by Battelle to digitize observed trips and code roadway functional classes. We developed models of average link travel speeds based on neighborhood, functional class, and link distance. Initial results confirm Jan, Horowitz and Peng's observation; we are only able to obtain 15 percent coverage from the least time path using an 80 percent overlap threshold. The least distance path offers 12 percent coverage at the same overlap threshold.

6.2.2 Data Collection Refinements

This thesis has shown that attitudinal questions regarding wayfinding can be helpful in explaining route choice behavior. MIT is working with the University of California-Santa Barbara to develop more effective survey instruments from which measures of network knowledge can be estimated. Currently, the team (MIT/UCSB, 2001) is conducting a small-scale evaluation of one such instrument through a web-based survey. Survey instruments are shown in Appendix F. Respondents are recruited from a convenience sample known by the research team. We anticipate that MPOs seeking to upgrade their modeling capabilities will wish to include wayfinding questions such as those presented in Appendices B and F.

A related effort involves the development of a handheld device for travel and attitudinal data collection. Global Positioning System receivers allow the collection of more complete and detailed route choice data, which should result in better quality route choice models. The use of GPS receivers in conjunction with electronic travel diaries is becoming more cost-effective and wide-spread, as Battelle (1997), Wolf et al. (2000), Wolf(2001), GeoStats (2001) and McNally (2001) illustrate. The MIT team is developing a combination personal digital assistant and GPS receiver prototype that will not only combine coordinate logging with an electronic travel diary, but also provide customized stated-preference exercises for participants. For example, the SP exercises may ask the subject to estimate the travel time and distance between two familiar (that is, frequently-visited) destinations, between two unfamiliar destinations, or between a familiar and an unfamiliar location. Clearly, we expect traveler perceptions to be strongly influenced by familiarity and experience. This effort is described in more detail in MIT/UCSB (2000).

6.2.3 Network Modeling and Turn Penalties

The coverage results presented in Section 5.1 highlights the deficiencies of current path generation algorithms. Clearly, a better behavioral understanding of the path selection process is needed. We observed that some travelers appeared to switch between corridors that were identified by the path generation algorithms presented. Algorithms that are able to incorporate these corridor shifts would therefore have better coverage. These algorithms might be developed by examining whether certain landmarks or subordinate "anchor points" can be found at these switching locations. Another approach involves creating new paths from an existing path set in a manner analogous to genetic mutation. That is, in the same way that mutations occur when overlapping DNA strands exchange genetic material, new paths may be created by exchanging link segments with overlapping paths.

Also, current metropolitan transportation planning programs may have limitations in their capabilities to generate and skim paths. For example, TransCAD offers the capability to sum a link variable over all links in a path, or at "transfer" points, that is, when another link variable changes value. During this research effort, we were often aware of the effects of turns (particularly left turns) and facility continuity on route choice. Adler, Blue and Wu (1999) present a route choice model that optimizes a function of link impedance and turn complexity. Clearly, the number and type of turns involved in a path would be a useful explanatory variable for path utilities. While TransCAD allows turn penalties to be applied in the process of selecting a shortest path, there is no capability to easily examine the effect of these turn penalties afterwards. That is, it is difficult to skim turn penalties. Bachu (2000) is another researcher expressing the desire for this capability.

At present, a heuristic may be used to estimate the number of turns. Very small generic turn penalties having a high least common multiple may be used. For example, penalties of 0.0097 and 0.0101 minutes may be given to right and left turns, respectively. Notice that 97 and 101 are prime, so their least common multiple is their product. Total path impedance is the sum of the link impedances plus the turn penalties. Since the number of right and left turns in a path will be integer, it should be possible to deduce the number of right and left turns from the sum of the turn penalties. Of course, different penalty values will have to be selected if paths typically contain more than 100 links.

Additionally, the ability to address roadway continuity may require both new data structures and path-building algorithms. Drivers appear not to perceive transportation facilities as individual links, but as collections of links into continuous facilities such as streets, highway routes or freeways. These facilities may be identified by a common name (e.g., Mass. Ave.) or route number, but these may also change at town or state boundaries. Travelers are able to distinguish between when a route number or name change represents a new facility – perhaps if a turn or exit is involved – and when a change is more “cosmetic” – that is, the road continues more or less straight, but its name has changed.

Travelers may perceive a collection of links to be a single facility even if they make multiple turns, or merge and diverge. Serpentine facilities such as Storrow Drive confound efforts to model facility continuity by using generic turn penalties, that is, penalties based on change in compass direction, as described in Section 4.2.3. (Recall that we set penalties for freeways and expressways to zero regardless of turn direction.)

TransCAD offers a route system data structure representing collections of often – but not necessarily – continuous links. This structure is traditionally used for bus routes, which have a well-defined identity and transfer structure. Some thought could be given to how this or a similar, new data structure could be used to represent what travelers perceive as a single facility.

6.2.4 Dynamic Models

The models presented in this thesis are static; they deal with habitual work travel and use a traditional planning network. Such a network implicitly has only two time periods: the peak hour for which modeling is conducted, and whenever free-flow conditions are relevant. Survey respondents reported selecting their route based on their knowledge of how travel times vary by time of day, day of week, season of year (summer, school vacation, Red Sox games). This reality and the popularity of dynamic traffic assignment programs to test ITS installations suggests great benefit could be realized by investigating how travelers perceive and remember dynamic aspects of travel.

6.2.5 Awareness of Transit

This research project initially hoped to examine transit route choice alongside automotive route choice. As work progressed, it became obvious that knowledge of public transportation is much more complicated because it encompasses many more dimensions of awareness – the route a bus or train may travel, the location of stops and stations, the fare structure and transfer policies, and the schedule are all considerations. More data needs to be collected from surveys targeted to public transportation passengers. Further, the tendency of transit networks to evolve slowly and its low share of urban trips may complicate assembling a sample of novice riders for sufficient statistical accuracy. (There are some counter-examples of the former concern; Madison, Wisconsin, and San Juan, Puerto Rico, have both recently changed their route structures to use a transfer center concept.)

A
References

Mohamed A. Abdel-Aty, Ryuichi Kitamura and Paul P. Jovanis (1996) “Investigating Effect of Advanced Traveler Information on Commuter Tendency to Use Transit.” *Transportation Research Record*, **1550**, 66.

Mohamed A. Abdel-Aty and Paul P. Jovanis (1998) “Modeling Incident-Related Routing Decisions Including the Effect of Traffic Information Using a Nested Logit Structure.” Presented at the 77th Annual Meeting of the Transportation Research Board, Washington DC, January 1998.

Mohamed A. Abdel-Aty, Kenneth M. Vaughn, Paul P. Jovanis, Ryuichi Kitamura and Fred L. Mannering (1994) “Impact of Traffic Information on Commuters’ Behavior: Empirical Results from Southern California and their Implications for ATIS.” *The Proceedings of the 1994 Annual Meeting of IVHS America*, 823-830.

Jeffrey L. Adler, Victor J. Blue and Tzu-Li Wu (1999) “Assessing Network and Driver Benefits from Bi-Objective In-Vehicle Route Guidance.” Presented at the 78th Annual Meeting of the Transportation Research Board, Washington DC, January 1999.

Jeffrey L. Adler and Michael G. McNally (1994) “In-Laboratory Experiments to Investigate Driver Behavior Under Advanced Traveler Information Systems.” *Transportation Research*, **2C**, 149-164.

Jeffrey L. Adler, Wilfred W. Recker and Michael G. McNally (1993) “A Conflict Model and Interactive Simulator (FASTCARS) for Predicting En-Route Driver Behavior in Response to Real-Time Traffic Condition Information.” *Transportation*, **20**, 2, 83-106.

Kazi Ahmed (1999) *Modeling Drivers’ Acceleration and Lane Changing Behavior*. MIT Ph.D. Dissertation.

Ravindra K. Ahuja, Thomas L. Magnanti and James B. Orlin (1993) *Network Flows: Theory, Algorithms, and Applications*. Englewood Cliffs, NJ: Prentice Hall.

J. Almond (1967) “Traffic Assignment with Flow-Dependent Journey Times” In *Vehicular Traffic Science*, Proceedings from the 3rd International Symposium on the Theory of Traffic Flow. L.C. Edie, R. Herman and R. Rothery, eds. American Elsevier, New York.

Richard Arnott, André de Palma and Robin Lindsey (1991) “Does Providing Information to Drivers Reduce Traffic Congestion?” *Transportation Research*, **25A**, 309-318.

J. A. Azevedo, M. E. O. Santos Costa, J. J. E. R. Silvestre Madeira and E. Q. Vieira Martins (1993) "An Algorithm for the Ranking of Shortest Paths." *European Journal of Operational Research*, **69**, 97-106.

Prashanth Bachu (2000) Personal communication.

Battelle (1997) "Global Positioning Systems for Personal Travel Surveys: Lexington Area Travel Data Collection Test." Final Report to Federal Highway Administration, 15 September 1997.

Shlomo Bekhor (2001) "Comparison of Alternative Route Choice Models." Presented at MIT Route Choice Seminar, 16 February 2001.

R. Bellman and R. Kalaba (1968) "On *k*th Best Policies." *Journal of the Society for Industrial and Applied Mathematics*, **10**, 582-588.

Moshe E. Ben-Akiva and Bruno Boccarà (1995) "Discrete Choice Models with Latent Choice Sets" *International Journal of Research in Marketing*, **12**, 9-24.

Moshe E. Ben-Akiva, M. J. Bergman, Andrew J. Daly and Rohit Ramaswamy (1984) "Modelling Inter Urban Route Choice Behavior." Ninth International Symposium on Transportation and Traffic Theory. VNU Science Press. 299-330.

Moshe E. Ben-Akiva and Michel Bierlaire (1999) "Discrete Choice Methods and Their Applications to Short Term Travel Decisions." in *Handbook of Transportation Science*, Randolph W. Hall, ed.

Moshe E. Ben-Akiva, Michel Bierlaire, Haris N. Koutsopoulos and Rabi Mishalani (1998) "DYNAMIT: A Simulation-Based System for Traffic Prediction and Guidance Generation." Presented at TRISTAN III, San Juan, Puerto Rico.

Moshe E. Ben-Akiva and Denis Bolduc (1996) "Multinomial Probit with a Logit Kernel and a General Parametric Specification of the Covariance Structure." Working paper.

Moshe E. Ben-Akiva, Jon Bottom and M. Scott Ramming (2000) "Route Guidance and Information Systems." Prepared for *Journal of Systems and Control Engineering*, Special Issue on Road Traffic Modelling and Control. Markos Papageorgiou and Christina Diakaki, eds.

Moshe E. Ben-Akiva and John L. Bowman (1995) *Activity Based Disaggregate Travel Demand Model System with Daily Activity Schedules*. Unpublished paper, Massachusetts Institute of Technology.

Moshe E. Ben-Akiva and John L. Bowman (1998) "Integration of an Activity Based Model System and a Residential Location Model." *Urban Studies*, **35**, 7, 1131. Special Issue on Transportation and Land Use, Alex Anas, ed..

Moshe E. Ben-Akiva, John L. Bowman and Dinesh A. Gopinath (1996) "Travel Demand Model System for the Information Era." *Transportation*, **23**, 241-266.

Moshe E. Ben-Akiva, John L. Bowman, M. Scott Ramming and Joan Walker (1998) "Behavioral Realism in Urban Transportation Planning Models." Presented at Transportation Models in the Policy-Making Process: Uses, Misuses and Lessons for the Future; a symposium on the problems of transportation analysis and modeling in the world of politics, in memory of Greig Harvey, Pacific Grove, California, March 4-6, 1998.

Moshe E. Ben-Akiva, André de Palma and Isam Kaysi (1991) "Dynamic Network Models and Driver Information Systems." *Transportation Research*, **25A**, 5, 251-266.

Moshe E. Ben-Akiva and Steven R. Lerman (1985) *Discrete Choice Analysis: Theory and Application to Travel Demand*. MIT Press, Cambridge, MA.

Moshe E. Ben-Akiva, Daniel McFadden, Makoto Abe, Ulf Böckenhold, Denis Bolduc, Dinesh Gopinath, Takayuki Morikawa, Venkatram Ramaswamy, Vithala Rao, David Revelt and San Steinberg (1997) "Modeling Methods for Discrete Choice Analysis." *Marketing Letters*, **8**, 3, 273-286.

Moshe E. Ben-Akiva and M. Scott Ramming (1998) "Lecture Notes: Discrete Choice Models of Traveler Behavior in Networks." Prepared for Advanced Methods for Planning and Management of Transportation Networks. Capri, Italy. 25 May 1998.

Moshe E. Ben-Akiva, M. Scott Ramming and Joan Walker (1999) "Improving Behavioral Realism of Urban Transportation Models Through Explicit Treatment of Individual's Spatial Ability." Presented at the European Science Foundation/US National Science Foundation *Conference on Social Change and Sustainable Transport*. Berkeley, California. 11 March 1999.

Moshe E. Ben-Akiva, Joan Walker, Adriana T. Bernardino, Dinesh A. Gopinath, Taka Morikawa and Amalia Polydoropoulou (1998) "Integration of Choice and Latent Variable Models." Presented at the AMA ART Forum, Keystone, CO.

Michel Bierlaire, Kay W. Axhausen and G. Abay (2001) "The Acceptance of Modal Innovation: The Case of Swissmetro." Presented at the 1st Swiss Transport Research Conference, Monte Verità / Ascona, Switzerland, 1-3 March 2001.

Denis Bolduc (1999) "A Practical Technique to Estimate Multinomial Probit Models in Transportation." *Transportation Research*, **33B**, 1, 63-79.

Peter Bonsall and M. Joint (1991) *Driver Compliance with Route Guidance: The Evidence and Its Implications*. 47-59. Society of Automotive Engineers, Warrendale, PA.

Jon Bottom (2000) *Consistent Anticipatory Route Guidance*. MIT Ph.D. Dissertation.

Piet H. L. Bovy and Eliahu Stern (1990) *Route Choice: Wayfinding in Transport Networks*. Studies in Industrial Organization, Kluwer Academic Publishers.

Daniel Brand (1998) "Applying Benefit/Cost Analysis To Identify and Measure the Benefits of Intelligent Transportation Systems." *Transportation Research Record*, **1651**, 23.

J. R. Busemeyer and J. T. Townsend (1993) "Decision Field Theory: A Dynamic-Cognitive Appraoch to Decision Making in an Uncertain Environment." *Psychological Review*, **100**, 3, 432-459.

Caliper Corporation (1996) *TransCAD User's Guide*. Version 3.0. Newton, MA.

Caliper Corporation (1996) *Travel Demand Modeling with TransCAD 3.0*. Newton, MA

Caliper Corporation (1998) *GISDK Programmer's Guide* with *Supplement for GIS+ and TransCAD*. Version 3.0. Newton, MA.

C. F. Camerer (1989) "An Experimental Test of Several Generalized Utility Theories." *Journal of Risk and Uncertainty*, **2**, 61-104.

Chris Caplice and Hani S. Mahmassani (1992) "Aspects of Commuting Behavior: Preferred Arrival Time, Use of Information and Switching Propensity." *Transportation Research*, **26A**, 5, 409-418.

Ennio Cascetta (2001) *Transportation Systems Engineering: Theory and Methods*. Applied Optimization Series, **49**. Norwell, MA: Kluwer Academic Publishers.

Ennio Cascetta, Agostino Nuzzolo, Francesco Russo and Antonino Vitetta (1996) "A Modified Logit Route Choice Model Overcoming Path Overlapping Problems: Specification and Some Calibration Results for Interurban Networks." *Transportation and Traffic Theory*. Proceedings from the Thirteenth International Symposium on Transportation and Traffic Theory, Lyon, France. J.B. Lesort, ed. Pergamon.

Ennio Cascetta and Andrea Papola (1998) "Implicit Availability/Perception Logit Models for Route Choice in Transportation Networks." Presented at the 8th *World Conference on Transport Research*, Antwerp.

Ken Cervenka (1999) Personal communication.

Ismail Chabini, Michael Florian and E. La Saux (1996) "Parallel and Distributed Computation of Shortest Routes and Network Equilibrium Models." Working paper.

Mingyuan Chen and Attahiru Sule Alfa (1991) "Algorithms for Solving Fisk's Stochastic Traffic Assignment Model." *Transportation Research*, **25B**, 6, 405-412.

Loveday Conquest, Jan H Spyridakis, Mark Haselkorn and Woodrow Barfield (1993) "The Effect of Motorist Information on Commuter Behavior: Classification of Drivers into Commuter Groups." *Transportation Research*, **1C**, 183-201

Carlos F. Daganzo (1977) "Multinomial Probit and Qualitative Choice: A Computationally Efficient Algorithm." *Transportation Science*, **11**, 338-358.

Carlos F. Daganzo and Yosef Sheffi (1977) "On Stochastic Models of Traffic Assignment." *Transportation Science*, **11**, 3, 253-274.

Ann K. Deakin (1997) "Procedural Knowledge and Its Potential for Enhancing Advanced Traveler Information Systems." Presented at the 76th Annual Meeting of Transportation Research Board, Washington DC, January 1997.

Tomas de la Barra, B. Perez and J. Anez (1993) "Multidimensional Path Search and Assignment." *Proceedings of the 21st PTRC Summer Meeting*, 307-319.

Karine Delvert (1997) “Heterogenous Agents Facing Route Choice: Experienced versus Inexperienced Tripmakers.” Presented at IATBR ‘97, Austin, TX.

André de Palma and Fabrice Marchal (1998) “METROPOLIS: A Dynamic Simulation Model Designed for ATIS Applications.” Presented at the 8th World Conference on Transport Research, Antwerp, Belgium.

Robert B. Dial (1971) “A Probabilistic Multipath Traffic Assignment Algorithm which Obviates Path Enumeration.” *Transportation Research*, **5**, 2, 83-111.

E. W. Dijkstra (1959) “Note on Two Problems in Connection with Graphs.” *Numerical Mathematics*, **1**, 269-271.

S. E. Dreyfus (1969) “An Appraisal of Some Shortest Path Algorithms.” *Operational Research*, **17**, 395-412.

East-West Gateway Coordinating Council (1997) “Travel Demand Model Documentation: A Report on the Travel Demand Model for the St. Louis Region, Including Changes Made for the 1998-2000 TIP, and a Compilation of Reference Files.” St. Louis, MO, November 1997.

Max J. Egenhofer and Reginald G. Golledge, eds. (1998) *Spatial and Temporal Reasoning in Geographic Information Systems*. Oxford Press, New York, NY.

A. Edland and Ola Svenson (1993) “Judgement and Decision Making Under Time Pressure: Studies and Findings.” In *Time Pressure and Stress in Human Judgement and Decision Making*, O. Svenson and J. A. Maule, eds. New York: Plenum Press. 27-40.

Caroline S. Fisk (1980) “Some Developments in Equilibrium Traffic Assignment.” *Transportation Research*, **14B**, 3, 243-255.

Scott M. Freundschuh (1992) “Is There a Relationship Between Spatial Cognition and Environmental Patterns?” *Lecture Notes in Computer Science*, **639**, 288-304. New York: Springer-Verlag.

Nathan Gale, Reginald G. Golledge, James W. Pellegrino and Sally Doherty (1990) “The Acquisition and Integration of Route Knowledge in an Unfamiliar Neighborhood.” *Journal of Environmental Psychology*, **10**, 3-25.

Tommy Gärling (1998) Working paper.

Tommy Gärling, Anders Böök, Erik Lingberg and Constantino Arce (1990) “Is Elevation Encoded in Cognitive Maps?” *Journal of Environmental Psychology*, **10**, 341-351.

Tommy Gärling and Margareta Friman (1998) “Psychological Principles of Residential Choice.” Draft chapter prepared for *Residential Environments: Choice, Satisfaction and Behavior*, J. Aragones, G. Francescato and Tommy Gärling, eds. Westport, CT: Greenwood.

GeoStats (2001) “Introduction to the GeoStats GeoLogger.” Product brochure.

Reginald G. Golledge, with R. Jacobson, R. Kitchin and M. Blades (1999) “Rethinking Theories of Blind

People's Spatial Abilities." (submitted)

Reginald G. Golledge, ed. (1999) *Wayfinding Behavior: Cognitive Mapping and Other Spatial Processes*. Johns Hopkins University Press, Baltimore, MD.

Reginald G. Golledge (1997) "Dynamics and ITS: Behavioral Responses to Information Available from ATIS." Presented at the International Association for Traveler Behavior Research conference, September 21-25, 1997, Austin, TX.

Reginald G. Golledge (1993) "Geographical Perspectives on Spatial Cognition." In *Behavior and Environment: Psychological and Geographical Approaches*, Tommy Gärling and Reginald G. Golledge, eds. Elsevier Science Publishers.

Dinesh A. Gopinath (1995) *Modeling Heterogeneity in Discrete Choice Processes: Application to Travel Demand*. MIT Ph.D. Thesis.

Christian Gouriéroux and Alain Monfort (1996) *Simulation-Based Econometric Methods*. CORE Lectures, Oxford University Press.

Hague Consulting Group (1995) *ALOGIT Users' Guide*. Version 3.8. HCG Report 5001-1.

Vassilis A. Hajivassiliou, Daniel McFadden and Paul A. Ruud (1996) "Simulation of Multivariate Normal Rectangle Probabilities: Methods and Programs." *Journal of Economics*, **72**, 1/2, 85.

Randolph W. Hall (1996) "Route Choice and Advanced Traveler Information Systems on a Capacitated and Dynamic Network." *Transportation Research*, **4C**, 5, 289-306.

Bijun Han, Staffan Algers and Leonid Engleson (2001) "Accommodating Drivers' Taste Variation and Repeated Choice Correlation in Route Choice Modeling by Using the Mixed Logit Model." Presented at the 80th Annual Meeting of the Transportation Research Board, Washington DC, January 2001.

Susan Hanson and Geraldine Pratt (1995) *Gender, Work and Space*. London: Routledge.

Yaser E. Hawas, Hani S. Mahmassani, R. Taylor, Athanasios Ziliaskopoulos, G.-L. Ghang and Srinivas Peeta (1997) "Development of DYNASMART-X Software for Real-Time Dynamic Traffic Assignment," Technical Report ST067-85-Task E, Center for Transportation Research, The University of Texas at Austin.

Steven C. Hirtle and Tommy Gärling (1992) "Heuristic Rules for Sequential Spatial Decisions." *Geoforum*, **23**, 2, 227-238.

W. Hoffman and R. Pavley (1959) "A Method for the Solution of the *n*th Best Path Problem." *Journal of ACM*, **6**, 506-514.

Alan J. Horowitz (1997) "Intersection Delay in Regionwide Traffic Assignment: Implications of the 1994 Update of the Highway Capacity Manual." Presented at the 76th Annual Meeting of Transportation Research Board, Washington DC, January 1997.

Alan J. Horowitz and Sam Granato (2000) “Selection of a Traffic Control Strategy for Long-Range Travel Forecasting.” Presented at the 79th Annual Meeting of Transportation Research Board, Washington DC, January 2000, and submitted for publication in *Transportation Research Record*.

Yasunori Iida, Takamasa Akiyama and Takashi Uchida (1992) “Experimental Analysis of Dynamic Route Choice Behavior.” *Transportation Research*, **26B**, 17-32.

Indian Nation Council of Governments (1998) “INCOG’s Population Estimates.” http://www.incog.org/1998%20pop%20est.htm .

Oliver Jan, Alan J. Horowitz and Zhong-Ren Peng (2000) “Using GPS Data to Understand Variations in Path Choice.” Presented at the 79th Annual Meeting of Transportation Research Board, Washington DC, January 2000.

R. Jayakrishnan, Michael G. McNally and Arun G. Marar (1995) “Recursive Structure for Exact Line Probabilities and Expected Waiting Time in Multipath Transit Assignment.” *Transportation Research Record*, **1493**, 178-187.

Mithilesh Jha, Samer Madanat and Srinivas Peeta (1998) “Perception Updating and Day-to-Day Travel Choice Dynamics in Traffic Networks with Information Provision.” *Transportation Research*, **6C**, 3, 189.

Daniel Kahneman and Amos Tversky (1984) “Choices, Values and Frames.” *American Psychologist*, **39**, 341-350.

Daniel Kahneman and Amos Tversky (1979) “Prospect Theory: An Analysis of Decision under Risk.” *Econometrica*, **47**, 263-291.

S. Kaplan (1976) “Adaptation, Structure and Knowledge.” In *Environmental Knowing*, G. T. Moore and Reginald G. Golledge, eds. Stroudsburg, PA: Dowden, Hutchinsen and Ross.

Isam Kaysi (1992) *Framework and Models for the Provision of Real-Time Driver Information*. MIT Ph.D. Dissertation.

Aemal J. Khattak and Asad J. Khattak (1998) “A Comparative Analysis of Spatial Knowledge and En-Route Diversion Behavior Across Chicago and San Francisco: Implications for ATIS.” Presented at the 77th Annual Meeting of Transportation Research Board, Washington, DC, January 11-15, 1998.

Asad J. Khattak, Amalia Polydoropoulou, and Moshe E. Ben-Akiva (1996) “Modeling Revealed and Stated Pretrip Travel Response to Advanced Traveler Information Systems.” *Transportation Research Record*, **1537**, 46.

Asad J. Khattak, Joseph L. Schofer and Frank S. Koppelman (1995) “Effect of Traffic Information on Commuters’ Propensity to Change Route and Departure Time.” *Journal of Advanced Transportation*, **29**, 2, 193.

William Hing-Keung Lam (1999) “A Stochastic User Equilibrium Assignment Model for Congested Transit Networks.” *Transportation Research*, **33B**, 5, 351-368.

Carol A. Lawton, Stephanie I. Charleston and Amy S. Zieles (1996) “Individual- and Gender-Related Differences in Indoor Wayfinding.” *Environment and Behavior*, **28**, 2, 204.

Yu-Hsin Liu and Hani Mahmassani (1998) “Dynamic Aspects of Departure Time and Route Decision Behavior Under Advanced Traveler Information Systems (ATIS): Modeling Framework and Experimental Results.” Presented at the 77th Annual Meeting of the Transportation Research Board, Washington, DC, January 1998.

Tsippy Lotan (1997) “Effects of Familiarity on Route Choice Behavior in the Presence of Information.” *Transportation Research*, **5C**, 3/4, 225-243.

Hani S. Mahmassani and Yaser E. Hawas (1998) “A Hierarchical Distributed Computational Architecture for A Centralized Real-time Dynamic Traffic Assignment.” Presented at TRISTAN III, San Juan, Puerto Rico.

Hani S. Mahmassani and R Jayakrishnan (1991) “System Performance and User Response Under Real-Time Information in a Congested Traffic Corridor.” *Transportation Research*, **25A**, 5, 293-307.

Fred Mannering, Soon-Gwan Kim, Linda Ng and Woodrow Barfield (1995) “Travelers’ Preferences for in-Vehicle Information Systems: An Exploratory Analysis.” *Transportation Research*, **3C**, 6, 339.

E. Q. Vieira Martins (1984) “An Algorithm for Ranking Paths That May Contain Cycles.” *European Journal of Operational Research*, **18**, 123-130.

MIT Personnel Office (1997) “FY ’98 (October 1997) Employee Headcounts.” Spreadsheet provided to MIT Planning Office.

MIT/UCSB (2000) “Work Program for NSF Proposal BCS-9986475: Individuals’ Spatial Abilities and Behavior in Transportation Networks.” Submitted to the National Science Foundation.

MIT/UCSB (2001) “MIT/UCSB Travel Choice and Wayfinding Survey.” http://web.mit.edu/its/wayfinding/ .

Massachusetts Turnpike Authority (1998) “Toll Schedule: Class 1.” http://www.massturnpike.com/toll_mileage/class_1.html .

Daniel McFadden (1997) “Rationality for Economists.” Presented at the NSF Symposium on Eliciting Preferences, Berkeley, CA, July 29, 1997.

Michael G. McNally (2001) Personal communication.

Metro (1994) *Portland Travel/Activity Survey*.

Metro (1998) Portland, OR. Personal communication.

Panos Michalopoulos (1991) “Vehicle Detection Video Through Image Processing: The AUTOSCOPE System.” *IEEE Transactions on Vehicular Technology*, **40**, 1, 21-34.

Pitu Mirchandani and Hossein Soroush (1987) “Generalized Traffic Equilibrium with Probabilistic Travel Times and Perceptions.” *Transportation Science*, **21**, 3, 133-152.

Kai Nagel, C. L. Barrett and M. Rickert (1996) *Parallel Traffic Microsimulation by Cellular Automata and Application for Large-Scale Transportation Modeling*. Los Alamos National Laboratory.

Michael O'Neill (1991) "A Biologically Based Model of Spatial Cognition and Wayfinding." *Journal of Environmental Psychology*, **11**, 299-320.

New York Metropolitan Transportation Council (1997) "Transportation Models and Data Initiative: Final Report: Interim Analysis Method (IAM) Model." July 17, 1997.

Sang Nguyen and Stephano Pallottino (1994) *A Modeling Framework for the Passenger Assignment on a Transport Network with Time Tables*. Montréal: Université de Montréal, Centre de Researche sur les Transports.

Sang Nguyen and Stephano Pallottino (1985) *Equilibrium Traffic Assignment for Large Scale Transit Networks*. Montréal: Université de Montréal, Centre de Researche sur les Transports.

Agostino Nuzzolo and Francesco Russo (1998) "Departure Time and Path Choice Models for Intercity Transit Assignment." in *Travel Behavior Research: Updating the State of Play*. New York: Elsevier.

Juan de Dios Ortúzar and Luis G. Willumsen (1994) *Modelling Transport*, 2nd ed. Chichester, UK: Wiley.

Andrea Papola (2000) "Some Development of the Cross-Nested Logit Model." *Proceedings of the 9th IATBR Conference*, Gold Coast, July 2000.

Thomas R Parish (1994) *Case Studies of Market Research for Three Transportation Communication Products*. USDOT, Volpe National Transportation Systems Center.

Dongjoo Park and Laurence R. Rilett (1997) "Identifying Multiple and Reasonable Paths in Transportation Networks: A Heuristic Approach." *Transportation Research Record* **1607**, 31-37.

A. Emily Parkany (1993) *Using Vehicle to Roadside Communication Data for Incident Detection*. MIT Masters Thesis.

Scott Peterson (1998) Personal communication.

John Polak and Peter Jones (1993) "The Acquisition of Pre-Trip Information: A Stated Preference Approach." *Transportation*, **20**, 2, 179-198.

M. Pollack (1961) "The *k*th Best Route Through a Network." *Operational Research*, **9**, 578-580.

Amalia Polydoropoulou and Moshe E. Ben-Akiva (1998) "Effect of Advanced Traveler Information Systems (ATIS) on Travelers' Behavior." Presented at the 77th Annual Meeting of the Transportation Research Board, Washington, DC, January 1998.

Amalia Polydoropoulou, Moshe E. Ben-Akiva, Asad J. Khattak and Geoffrey Lauprête (1996) "Modeling Revealed and Stated En-Route Travel Response to Advanced Traveler Information Systems." *Transportation Research Record*, **1537**, 38.

Joseph N. Prashker and Shlomo Bekhor (1998) "Investigation of Stochastic Network Loading

Procedures." *Transportation Research Record*, **1645**, 94. .

Viplav Putta (1999) Personal communication.

M. Scott Ramming (1994) *A Consumption-Based Accessibility Index of Transportation and Land Use*. MIT Masters Thesis.

M. Scott Ramming (1999) *Mode Use Characteristics of MIT Faculty, Staff and Student Commuters*. Center for Transportation Studies report prepared for the MIT Transportation and Parking Committee.

Rand McNally (1994) Connecticut, Massachusetts and Rhode Island Regional Map.

Joseph L. Schofer, Frank S. Koppelman and William A. Charlton (1997) "Perspectives on Driver Preferences for Dynamic Route Guidance Systems." *Transportation Research Record*, **1588**, 26.

Kelley Scott, Glarycelis Pabón-Jiménez and David H. Bernstein (1997) "Finding Alternatives to the Best Path." Presented at the 76th Annual Meeting of the Transportation Research Board, Washington, DC, January 1997.

Yosef Sheffi (1985) *Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods*. Englewood Cliffs, NJ: Prentice-Hall.

Yosef Sheffi and Warren B. Powell (1982) "An Algorithm for the Equilibrium Assignment Problem with Random Link Times." *Networks*, **12**, 2, 191-207.

R. D. Shier (1979) "On Algorithms for Finding the *k* Shortest Paths in a Network." *Networks*, **9**, 195-214.

T. R. Smith, James W. Pellegrino and Reginald G. Golledge (1982) "Computational Process Modeling of Spatial Cognition and Behavior." *Geographical Analysis*, **14**, 305-325.

Heinz Spiess and Michael Alexander Florian (1989) "Optimal Strategies: A New Assignment Model for Transit Networks." *Transportation Research*, **23B**, 2, 83-102.

Statistics Canada (1999) "Population of Census Metropolitan Areas." Compiled from CANSIM Matrices 9248-9249. http://www.statcan.ca/english/Pgdb/People/Population/demo05.htm or http://www.statcan.ca/francais/Pgdb/People/Population/demo05_f.htm .

Eliahu Stern (1998) "Choice Behavior in Congestion Situation: Modeling and Research Needs." In *Theoretical Foundataions of Travel Choice Modeling*, T. Gärling, T. Laitila and K. Westin, eds. Oxford: Pergamon Press, 178-200.

Eliahu Stern and David Leiser (1988) "Levels of Spatial Knowledge and Urban Travel Modeling." *Geographical Analysis*, **20**, 2, 140-155.

P. Stopher (1980) "Captivity and Choice in Travel-Behavior Models." *Transportation Journal of ASCE*, **106**, 427-435.

Ola Svenson (1990) "Some Propositions for the Classification of Decision Situations." In *Contemporary Issues in Decision Making*, K. Borcherding, O. I. Larichev and D. M. Messick, eds. Amsterdam:

Elsevier.

Joffre Swait (2001) “Choice Set Generation within the Generalized Extreme Value Family of Discrete Choice Models.” *Transportation Research*, **35B**, 643-666.

Joffre Swait and Moshe E. Ben-Akiva (1986) “Analysis of the Effects of Captivity on Travel Time and Cost Elasticities.” In *Behavioral Research for Transport Policy*, 119-134. Utrecht, the Netherlands: VNU Science Press.

Daniëlle Timmermans and Ch. Viek (1992) “Multi-Attribute Decision Support and Complexity: An Evaluation and Process Analysis of Aided Versus Unaided Decision Making.” *Acta Psychologica*, **80**, 49-65.

Kenneth Train, David Revelt and Paul Ruud (1999a) “Mixed Logit Estimation Routine for Cross-Sectional Data.” http://elsa.berkeley.edu/Software/abstracts/train0196.html .

Kenneth Train, David Revelt and Paul Ruud (1999b) “Mixed Logit Estimation Routine for Panel Data.” http://elsa.berkeley.edu/Software/abstracts/train0296.html .

Amos Tversky (1977) “On the Estimation of Preferences: Descriptive and Prescriptive Considerations.” In *Conflicting Objectives in Decisions*, D. Bell, R. Kenney and H. Raiffa, eds. New York: Wiley.

Amos Tversky (1972) “Elimination by Aspects: A Theory of Choice.” *Psychological Review*, **79**, 281-299.

Amos Tversky and Craig R. Fox (1995) “Weighing Risk and Uncertainty.” *Psychological Review*, **102**, 269-283.

Amos Tversky and Daniel Kahneman (1992) “Advances in Prospect Theory: Cumulative Representation of Uncertainty.” *Journal of Risk and Uncertainty*, **5**, 297-323.

U.S. Bureau of the Census (1997) “MA-96-5 Estimates of the Population of Metropolitan Areas: Annual Time Series, July 1, 1991 to July 1, 1996.” Population Estimates Program, Population Division, U.S. Bureau of the Census, Washington, DC. http://www.census.gov/population/estimates/metro-city/ma96-05.txt .

U.S. Department of Commerce (1964) *Traffic Assignment Manual*. Bureau of Public Roads, Urban Planning Division, U.S. Department of Commerce, Washington, DC.

U.S. Department of Transportation (1986) “Urban Transportation Planning System (UTPS).” Program Documentation by the Urban Mass Transportation Administration and the Federal Highway Administration, Washington, DC.

Universal Publishing Co., Inc. (1997) Universal Atlas of Metropolitan Boston and Eastern Massachusetts. 30th ed.

Kenneth M. Vaughn, Mohamed A. Abdel-Aty, Ryuichi Kitamura, Paul P. Jovanis and H. Yang (1993) “Experimenting Analysis and Modeling of Sequential Route Choice Under ATIS in a Simple Traffic Network.” *PATH Research Report* 93-12.

Peter Vovsha and Shlomo Bekhor (1998) "The Link-Nested Logit Model of Route Choice: Overcoming the Route Overlapping Problem." *Transportation Research Record*, **1645**, 133.

Joan L. Walker (2001) *Extended Discrete Choice Models: Integrated Framework, Flexible Error Structures, and Latent Variables.* MIT Ph.D. Dissertation.

Joan Walker (1994) *Modeling Traveler Response to Traveler Information Systems: Laboratory Simulation of Information Searches Using Multimedia Technology*. MIT Masters Thesis.

J. G. Wardrop (1952) "Some Theoretical Aspects of Road Traffic Research." *Proceedings of the Institute of Civil Engineers*. Road Paper N36, 325-378.

Chieh-Hua Wen and Frank S. Koppelman (2001) "The Generalized Nested Logit Model." *Transportation Research*, **35B**, 627-641.

H. Williams and Juan de Dios Ortúzar (1982) "Behavioral Theories of Dispersion and Misspecification of Travel Demand Models." *Transportation Research*, **16B**, 3, 167-219.

Marianne Williamson (1992) *A Return To Love: Reflections on the Principles of A Course in Miracles*. Chapter 7, Section 3. Harper Collins.

Jean Wolf (2001) Personal communication.

Jean Wolf, Randall Guensler, Lawrence Frank and Jennifer Ogle (2000) "The Use of Electronic Travel Diaries and Vehicle Instrumentation Packages in the Year 2000 Atlanta Regional Household Travel Survey: Test Results, Package Configurations, and Deployment Plans." *Proceedings of the 9th IATBR Conference*, Gold Coast, July 2000.

Jean Wolf, Shauna Hallmark, Marcelo Oliveira, Randall Guensler and Wayne Sarasua (1999) "Accuracy Issues with Route Choice Data Collection Using GPS." Presented at the 78th Annual Meeting of the Transportation Research Board, Washington, DC, January 1999.

Jia Hao Wu, Michael Alexander Florian and Patrice Marcotte (1994) "Transit Equilibrium Assignment: A Model and Solution Algorithms." *Transportation Science*, **28**, 3, 193-203.

Tetsuo Yai, Seiji Iwakura and Shigeru Morichi (1997) "Multinomial Probit with Structured Covariance for Route Choice Behavior." *Transportation Research*, **31B**, 3, 195-207.

Jack Zhou and Reginald G. Golledge (2000) "An Analysis of Household Travel Behavior Based on GPS." *Proceedings of the 9th IATBR Conference*, Gold Coast, July 2000.

Athanasios Ziliaskopoulos (1994) *Optimum Path Algorithms on Multidimensional Networks: Analysis, Design, Implementation and Computational Experience*. University of Texas at Austin Ph.D. Dissertation Thesis.

B
1997 Transportation Survey for Employees Questionnaire / Code Book

MIT TRANSPORTATION SURVEY

Questionnaire for

Faculty and Staff

Codebook

MIT Planning Office

Thank you for your willingness to participate in the **1997 MIT Transportation Survey**. This survey should take about 20 to 30 minutes to complete. Your participation is **completely voluntary**, and all replies will be anonymous. The responses to the survey will be held in the strictest confidence and only be used for MIT purposes. You may skip any question you do not want to answer, and you may stop at any time. By providing your frank answers, you're giving the Institute the ability to better meet your needs and to improve your quality of life. There is a **glossary** at the end of this survey to explain any terms that may be unfamiliar.

obsno = *respondent identifier (1-1381)*

B.1 Background

1. What is the ZIP Code for your home? **zip?**

2. In which building do you work? **bldg?**

(You may use the following codes as the prefix for your building: **area**
1 - M - Main Campus
2 - W - West Campus
3 - E - East of Ames Street
4 - N - North of Railroad Tracks
5 - NE - Northeast
6 - NW - Northwest
7 - WW - West West
8 - LIN - Lincoln Lab
9 - OCR - Off-campus)

3. What is your status at MIT? **status**

a) **1** faculty b) **2** administrative staff c) **3** other academic staff
d) **4** support staff e) **5** research staff f) **6** service staff
g) **1** medical staff
h) **8** other (please specify)

4. With which department or program are you affiliated? **deptcode**

5. Which best describes your status? **timestat**

a) **1** full-time b) **2** part-time c) **3** visiting d) **4** retired

B.2 Your Commuting Patterns

1. In the past seven days, how many days did you come to MIT? **dayscame**
a) **0** none b) ❑ 1 c) ❑ 2 d) ❑ 3 e) ❑ 4 f) ❑ 5 g) ❑ 6 h) ❑ 7

2. Of these, how many days did you come to MIT over the past weekend? **wkendcam**
a) **0** none b) ❑ 1 c) ❑ 2

3. During the past seven days, how many days did you run errands as you came to MIT from home? **erndto**
a) **0** none b) ❑ 1 c) ❑ 2 d) ❑ 3 e) ❑ 4 f) ❑ 5 g) ❑ 6 h) ❑ 7

4. During the past seven days, how many days did you leave campus during the middle of your day? **erndmid**
a) **0** none b) ❑ 1 c) ❑ 2 d) ❑ 3 e) ❑ 4 f) ❑ 5 g) ❑ 6 h) ❑ 7

5. During the past seven days, how many days did you run errands as you left MIT for home? **erndleav**

a) **0** none b) ❑ 1 c) ❑ 2 d) ❑ 3 e) ❑ 4 f) ❑ 5 g) ❑ 6 h) ❑ 7

6. What were your reason(s) for these extra trips in Questions 3 through 5? **(please check ALL that apply.) (1 = checked)**

a) **ernd_mwr** MIT work related
b) **ernd_owr** other work related
c) **ernd_ccr** care for children and other relatives
d) **ernd_ma** medical appointments
e) **ernd_vw** volunteer work
f) **ernd_opb** other personal business

7. At what time do you typically arrive at MIT? **arr_h** : **arr_min** **arr_ampm 1** a.m. **2** p.m.

8. In general, by how much does your arrival time vary? **(please check only ONE) arr_vary**

a) **1** not at all
b) **2** by no more than 15 minutes
c) **3** by no more than 30 minutes
e) **4** by no more than an hour
f) **5** by more than an hour, but only in unusual circumstances
g) **6** often by more than an hour
h) **7** it depends on my class or meeting schedule

9. At what time do you typically leave MIT? **dep_h** : **dep_min** **dep_ampm 1** a.m. **2** p.m.

10. In general, by how much does your departure time vary? **(please check only ONE) dep_vary**

a) **1** not at all
b) **2** by no more than 15 minutes
c) **3** by no more than 30 minutes
e) **4** by no more than an hour
f) **5** by more than an hour, but only in unusual circumstances
g) **6** often by more than an hour
h) **7** it depends on my class or meeting schedule

11. Does your job allow you to have a more flexible schedule? **flex**

a) **1** yes b) **2** no c) **3** not sure

12. Do you have a driver's license? **drivlic** a) **1** yes b) **0** no **(please skip to Question 14)**

13. If you drive a car, do you have an MIT parking permit? **(please check only ONE) parkperm**

a) **1** yes, a monthly permit
b) **2** yes, an occasional use permit
c) **3** no, I don't need one
d) **4** no, I couldn't afford one
e) **5** no, I couldn't get one
f) **6** I don't drive around MIT

14. Do you have a monthly MBTA pass? **tpass** a) **1** yes b) **0** no **(please skip to Question 17)**

15. If you have a monthly MBTA pass, which one do you have? **(please check only ONE) tpasszn**

a) **1** local bus
b) **2** subway
c) **3** combo
e) **4** combo plus
f) **5** commuter boat

commuter rail:
g) **6** zone 1A
h) **7** zone 1B
i) **8** zone 1
j) **9** zone 2
k) **10** zone 3
l) **11** zone 4
m) **12** zone 5
n) **13** zone 6
o) **14** zone 7
p) **15** zone 8
q) **16** zone 9

16. If you have an MBTA pass, about how many days during the last month did you ride the MBTA for *any trip* (not just coming to MIT)? **tpassday** days

17. How many days during the past seven days did you work at home before coming into MIT, or after leaving MIT? **wkhmplus**

a) **0** none b) ❑ 1 c) ❑ 2 d) ❑ 3 e) ❑ 4 f) ❑ 5 g) ❑ 6 h) ❑ 7

18. How many days during the past seven days did you work at home *instead of* coming into MIT? **wkhminst**

a) **0** none b) ❑ 1 c) ❑ 2 d) ❑ 3 e) ❑ 4 f) ❑ 5 g) ❑ 6 h) ❑ 7

19. Would you say that compared to a year ago ... **(please check only ONE) wkhmcomp**

a) **1** you work at home *more* now than a year ago
b) **2** you work at home *less* now than a year ago
c) **3** you work at home about the same amount
d) **4** you can’t compare because you weren’t at MIT a year ago

20. If you work at home, do you connect via computer to MIT networks? **(please check only ONE) connect**

a) **1** yes
b) **2** no, I don’t connect to MIT networks **(please GO ON to Section C)**
c) **3** no, I don’t have a computer at home **(please GO ON to Section C)**
d) **4** no, I don’t work from home **(please GO ON to Section C)**

21. If you connect to MIT networks from home, what are your reasons for connecting? **(please check ALL that apply) (1 = checked)**

a) **con_em** to send and receive email
c) **con_ds** to access a departmental server
b) **con_ath** to access Athena
d) **con_imf** to access institute mainframes

B.3 Your Primary Way of Commuting to MIT

1. Think about your primary way of commuting to MIT. Which mode or modes do you use as you come from home to MIT? (If you vary your commute depending on the season, consider how you came to MIT the past seven days.)

Place a 1 by the mode you use first during your trip, a 2 by the one you use second (if any), and so on. (Don't count walking between vehicles. If you walk all the way from home to MIT, place a 1 by "walk all the way.")

Example 1. If most of the time, you bicycle to MIT, lock your bike at an outside rack, and then walk to your office, place a 1 by "bicycle." Don't place a 2 by "walk all the way."

Example 2. If you drive all the way to MIT and drop off a passenger along the way, place a 1 by "by car; driving, with passengers" and a 2 by "by car; driving alone."

Example 3. If someone drops you off at a commuter rail station, then you take the commuter rail to North Station, then the Green Line and the Red Line to Kendall Square, and then walk to your office, place a 1 by "by car; as a passenger," a 2 by "commuter rail," and a 3 by "subway."

Don't record how you get to MIT on days when you don't travel by your primary way.

Sequence in which you use this mode	Mode
seq_bike	bicycle
	bus
seq_tbus	MBTA bus
seq_tbs2	a second MBTA bus
seq_safe	SafeRide
seq_m2	M2 / LMA / medical shuttle
seq_lls	Lincoln Lab shuttle
seq_ws	Wellesley shuttle
seq_tma	Charles River TMA shuttle
	by car
seq_sov	driving alone
seq_hovd	driving, with passengers
seq_hovp	as a passenger
seq_boat	commuter boat, ferry
seq_cr	commuter rail
seq_mcyc	motorcycle
seq_sub	subway
seq_oth	other (please specify ________________________________)
seq_walk	walk all the way

2. How long does it typically take you to come to MIT this way? (Count the time from when you leave home to when you get to your office, lab or class) **tt_tot** minutes

3. Of your commuting time to MIT, how much do you spend ...
a) walking **tt_walk** minutes
b) waiting for vehicles **tt_wait** minutes
c) in vehicles **tt_ride** minutes

4. Estimate your **monthly** cost of commuting to and from MIT. Think about all your expenses, such as gas, parking, insurance, tolls, fares, transit passes, and wear and tear. $ **tc**

5. How satisfied are you with this way of commuting? Please rate your level of satisfaction with the following aspects of your trip:

	Very Dissatisfied	Dissatisfied	Neutral	Satisfied	Very Satisfied
a) **sat_tt** Speed / time	1	2	3	4	5
b) **sat_tc** Cost	1	2	3	4	5
c) **sat_conv** Convenience	1	2	3	4	5
d) **sat_flex** Flexibility / can make stops	1	2	3	4	5
e) **sat_priv** Privacy	1	2	3	4	5
f) **sat_safe** Safety	1	2	3	4	5
g) **sat_rely** Reliability	1	2	3	4	5
h) **sat_prod** Productive use of time	1	2	3	4	5
i) **sat_env** Environmentally responsible	1	2	3	4	5
j) **sat_oall** Overall rating	1	2	3	4	5

6. If your commute involves traveling by car, what are some of the major roads you take? (In other words, how would you describe your route to a neighbor or colleague?)

____

7. If you share a car with others, are they ... **(please check ALL that apply) (1 = checked)**
a) **hov_fam** family c) **hov_nei** neighbors e) **hov_ocar** members of an organized carpool
b) **hov_room** d) **hov_coll** f) **hov_ovan**
roommates classmates/officemates/labmates members of an organized vanpool

8. How did you find out about this way of coming to MIT? **(please check ALL that apply) (1 = checked)**
a) **lrn_ask** asking friends, relatives or colleagues
b) **lrn_map** using a map
c) **lrn_ts** using transit schedules
d) **lrn_il** calling an information line (for example, the MBTA information line, 222-3200)
e) **lrn_expl** exploring the city during spare time
f) **lrn_try** trying to find your way by following signs or asking strangers when necessary
g) **lrn_expr** I have a lot of experience with the city from other jobs, residences, attending various events, etc.
h) **lrn_cr** I can't remember

9. How often do you come this way to MIT, compared to other ways you might come to MIT (for example using different modes, or different streets when driving)? **(please check only ONE) primfreq**

a) **5** all the time

b) **4** 80 percent of the time (4 days out of five) or more

c) **3** 50 to 80 percent of the time

d) **2** 25 to 50 percent of the time

e) **1** less than 25 percent of the time (for example, during a week, you use a different way each day)

B.4 Opinions about Your Current Commute

1. Below are some reasons people may have for choosing **how** they commute (that is, which modes or routes) and **when** they commute. For each reason, please say to what extent each reason affects your choice of mode in column 1, and your choice of departure time in column 2. If a reason is not applicable to your commute, check the "Not Applicable" column.

		howt_<letter>					**whent_<letter>**				
		1 To what extent does this affect **how** you commute?					2 To what extent does this affect **when** you commute?				
	Not Applicable	Not at All	Very Little	Little	Much	Very Much	Not at All	Very Little	Little	Much	Very Much
a) My child or children need to get to day care or school	0	1	2	3	4	5	1	2	3	4	5
b) I must help care for an elder relative	0	1	2	3	4	5	1	2	3	4	5
c) I must help care for a family member with special needs	0	1	2	3	4	5	1	2	3	4	5
d) I frequently carry many heavy items (for example, books, tools, equipment)	0	1	2	3	4	5	1	2	3	4	5
e) There is not much public transportation near where I live	0	1	2	3	4	5	1	2	3	4	5
f) Finding parking near my home is too difficult	0	1	2	3	4	5	1	2	3	4	5
g) Finding parking near MIT can be difficult	0	1	2	3	4	5	1	2	3	4	5
h) The cost of owning and operating a car is too expensive	0	1	2	3	4	5	1	2	3	4	5
i) A disability prevents me from driving	0	1	2	3	4	5	1	2	3	4	5
j) A disability prevents me from using public transportation	0	1	2	3	4	5	1	2	3	4	5
k) I try to make decisions that take the environment into account	0	1	2	3	4	5	1	2	3	4	5
l) It is very important that I get to MIT on time	0	1	2	3	4	5	1	2	3	4	5
m) I don't feel safe walking at night	0	1	2	3	4	5	1	2	3	4	5
n) I don't feel safe riding public transportation at night	0	1	2	3	4	5	1	2	3	4	5
o) I don't feel safe waiting for public transportation at night	0	1	2	3	4	5	1	2	3	4	5
p) I don't feel comfortable/safe using public transportation during the day	0	1	2	3	4	5	1	2	3	4	5
q) I need my time traveling between home and MIT to relax and organize my thoughts	0	1	2	3	4	5	1	2	3	4	5
r) Other people in my household need our car(s)	0	1	2	3	4	5	1	2	3	4	5
s) The final walk to my lab, office or class is too long	0	1	2	3	4	5	1	2	3	4	5
t) I consult radio, TV or telephone traffic or transit reports to avoid delays	0	1	2	3	4	5	1	2	3	4	5

2. Below are some reasons people may have for living where they do. For each reason, please say to what extent each reason affects your choice of where to live. If a reason is not applicable to your choice of where to live, check the "Not Applicable" column. **wherel_<letter>**

		To what extent does this affect your choice of where you live?				
	Not Applicable	Not at All	Very Little	Little	Much	Very Much
a) I like living in a bustling city	0	1	2	3	4	5
b) I like living in a quiet neighborhood	0	1	2	3	4	5
c) I did not have much time to search for housing	0	1	2	3	4	5
d) I cannot afford to pay much for housing	0	1	2	3	4	5
e) I want to live in a community with a good school system	0	1	2	3	4	5
f) I want to live close to MIT because I often work late	0	1	2	3	4	5
g) I want to live close to MIT so I don't have to commute as far	0	1	2	3	4	5
h) I chose my house so I can commute by my preferred means	0	1	2	3	4	5
i) My neighborhood is safer than other places I could live	0	1	2	3	4	5
j) The features of my home (for example, the view, hardwood floors, air conditioning, laundry, yard) are a large factor in why I chose to live where I do	0	1	2	3	4	5
k) I like to live near other people who are involved with MIT	0	1	2	3	4	5
l) I live where I do because of my spouse, family or roommate	0	1	2	3	4	5

B.5 Transportation Options

1. Which of the following do you have at home? **(please check ALL that apply) (1 = checked)**

a) □ a **computer**

a modem:

- b) **modem96** 9600 bps or slower
- c) **modem144** 14.4 Kbps
- d) **modem288** 28.8 Kbps or faster
- e) **modemuns** I'm not sure of the modem's speed

f) **eth_card** an ethernet card

g) □ a **fax** machine

h) **phone2** a second phone line at home

i) □ a **tether** account

2. Here are some statements about attitudes toward commuting options. Please indicate the extent to which you agree with these statements. **att_<letter>**

	Not Applicable	Disagree Strongly	Disagree	Neutral	Agree	Agree Strongly
a) I can be more productive working at home	0	1	2	3	4	5
b) I need to interact with other people at MIT daily	0	1	2	3	4	5
c) Most of my work can be done at home	0	1	2	3	4	5
d) Working at home gives me the flexibility I need to care for my family	0	1	2	3	4	5
e) I enjoy city driving	0	1	2	3	4	5
f) I like learning one reliable way to come to MIT and sticking with it	0	1	2	3	4	5
g) I like learning about all the options before I settle on my preferred way of commuting	0	1	2	3	4	5
h) I do not have space to work at home	0	1	2	3	4	5
i) In order to advance, it's important to be around campus so others can see how productive you are	0	1	2	3	4	5
j) I need a car for carrying groceries	0	1	2	3	4	5
k) I need a car for weekend trips	0	1	2	3	4	5
l) The Office of Parking and Transportation provides sufficient information about commute options	0	1	2	3	4	5
m) MIT does a good job helping people find matches for carpooling or vanpooling	0	1	2	3	4	5
n) Public transportation schedules should be made more available on campus	0	1	2	3	4	5
o) MIT should encourage people to log in from home	0	1	2	3	4	5
p) MIT should encourage people to use public transportation more	0	1	2	3	4	5
q) MIT should encourage people to walk, bike, skate, etc., more	0	1	2	3	4	5
r) If I had a reliable, inexpensive way of getting home in an emergency, I would take public transportation, carpool or vanpool more often	0	1	2	3	4	5
s) MIT should have more shuttles to important transportation centers (such as North Station)	0	1	2	3	4	5
t) MIT should have an on-campus daytime shuttle	0	1	2	3	4	5
u) I find public transportation schedules confusing	0	1	2	3	4	5
v) I often consult maps and/or public transportation schedules	0	1	2	3	4	5
w) I'd feel comfortable parking a car at MIT lots	0	1	2	3	4	5
x) I'd feel comfortable locking a bike at MIT racks	0	1	2	3	4	5
y) I know my way around the city well, and can easily find another route	0	1	2	3	4	5
z) I change the way I travel depending on the time of day or season of the year	0	1	2	3	4	5

3. Based on your needs, which of the following list of transportation investments would you most like to see MIT make? Please indicate **your top three choices** by placing a 1 in the blank by your top choice, a 2 by your second choice, and a 3 by your third choice. **ir_<code>**

a) **fbus** more frequent shuttle bus service
b) **sside** smoother sidewalks around MIT
c) **cpark** more convenient parking spaces
d) **lside** better landscaping around sidewalks
e) **brack** more bicycle racks
f) **secur** better security for parking lots and garages
g) **tsub** greater subsidies for MBTA passes
h) **block** distributed bicycle lockers and showers
i) **sroad** smoother roadways around MIT
j) **net** enhanced access to MIT computer networks
k) **bpath** separate bike and pedestrian paths
l) **m2** free access to the M2 / LMA shuttle
m) **sbus** shuttle buses to ____________________
n) **oth** other: ____________________

4. Now, considering the needs of **the whole MIT community** and the environment, which do you think are the most important transportation investments for MIT to make? Please indicate your top three choices by placing a 1 in the blank by your top choice, a 2 by your second choice, and a 3 by your third choice. **mr_<code>**

a) **fbus** more frequent shuttle bus service
b) **sside** smoother sidewalks around MIT
c) **cpark** more convenient parking spaces
d) **lside** better landscaping around sidewalks
e) **brack** more bicycle racks
f) **secur** better security for parking lots and garages
g) **tsub** greater subsidies for MBTA passes
h) **block** distributed bicycle lockers and showers
i) **sroad** smoother roadways around MIT
j) **net** enhanced access to MIT computer networks
k) **bpath** separate bike and pedestrian paths
l) **m2** free access to the M2 / LMA shuttle
m) **sbus** shuttle buses to ____________________
n) **oth** other: ____________________

B.6 Inter-City Travel

This section addresses trips you need to make **by air** for business related to MIT.

1. How many times a year do you need to make business trips by air? **airtrip**

(If you don't make any business trips by air, enter zero, and skip to section G)

2. Please think back to your **last business trip** when you flew somewhere outside of the Boston metropolitan area. Where did you park your car when you made this trip? **(please check only ONE)** **aircarpk**

a) **1** at the airport
b) **2** at a park and ride lot
c) **3** at MIT
d) **4** at home / I didn't drive / I don't have a car

3. How did you get between MIT and the airport? **(please check ALL that apply)**

a) **air_car** by personal car
b) **air_t** by MBTA
c) **air_pass** by a car driven by a friend, relative or associate
d) **air_shut** by a shuttle bus
e) **air_taxi** by taxi
f) **air_xmit** I didn't leave from MIT

B.7 Some Final Questions

The following questions are for classification purposes and assist in identifying certain trends. Your responses are confidential and will be held in the strictest confidence.

1. About how long have you lived in the Boston metro area? **live_bos** years

2. About how long have you worked at MIT? **work_mit** years

3. About how long have you lived at your current residence? **live_res** years

4. What is your **gender**? a) **1** female b) **2** male

5. What is your **age**?
a) **1** 16 or 17 years
b) **2** 18 to 20 years
c) **3** 21 to 24 years
d) **4** 25 to 29 years
e) **5** 30 to 39 years
f) **6** 40 to 49 years
g) **7** 50 years or older

6. Are you ... **marital**
a) **1** single, separated, divorced or widowed b) **2** married or in a long-term relationship

7. How many children live in your home? **children**

8. Do you ... **own_rent** a) **1** own your home b) **2** rent or lease your home, condominium or apartment

9. Which of the following categories would you use to describe yourself? **(please check ALL that apply) (1 = checked)**
a) **asian**
b) **black**
c) **hispanic**
d) **nat_am** Native American
e) **cauc** Caucasian
f) **oth_eth** Other (please specify ________________________)

10. Which category best describes your annual household **income**, before taxes?
a) **1** Less than $25,000
b) **2** $25,000 to $49,999
c) **3** $50,000 to $74,999
d) **4** $75,000 to $99,999
e) **5** $100,000 to $149,999
f) **6** $150,000 or more

11. Please give us any comments you may have about transportation at MIT or about this survey.

Thank you very much for your assistance with our research and planning efforts. Please fold this survey form in half, tape or staple it shut, and place it in any interdepartmental mail box.

The following additional variables were added to the survey dataset:

ring = a designation of residence community (based on **zip**) used by the City of Cambridge
1 Cambridge
2 Abutting communities (Arlington, Belmont, Boston, Brookline, Somerville and Watertown)
3 All other communities

modelbl = described the primary mode used to commute to MIT (based on **seq_** variables)

1	single-occupancy vehicle
2	HOV – in carpool as driver or passenger, or in organized vanpool
3	walk to transit
4	Park-and-Ride (drive to transit)
5	bicycle
6	motorcycle
7	walk (all the way)
8	other modes (jog, inline skate, etc.)

nwt = normalized weight based on **gender** and **status**
To expand to the whole population, multiply **nwt** by **8177/1381**.

B.8 Glossary

Boston metropolitan area (or the city, etc.)

The Boston metropolitan area includes Boston and all the cities and towns around it that are economically tied to it. Since you commute to MIT, wherever you live is considered part of the Boston metropolitan area.

carpool

A carpool is two or more people together in a car or other privately-owned vehicle.

Charles River TMA

MIT and several other employers in Cambridge participate in a Transportation Management Association (TMA) called the Charles River TMA to provide better travel options to their employees, and to comply with federal and state transportation and environmental policies. The TMA may provide assistance managing parking spaces, organizing car- and vanpools, and operating shuttle buses. The Charles River TMA runs a shuttle from near Tech Square and One Kendall Square to near the B.U. Bridge. The TMA shuttle is operated under contract by Paul Revere Transportation.

commute

For this survey, a commute refers to a trip from your home to MIT, or from MIT back home. You may stop off along the way at other places.

M2 / LMA / medical shuttle

Paul Revere Transportation operates a shuttle bus -- identifiable from its blue stripes -- for Harvard. The M2 route connects the Harvard Medical School in the Longwood Medical Area (LMA) to the main Harvard campus in Cambridge. This shuttle bus also stops at 77 Mass. Ave.

mode

Mode refers to your method of travel, and is often described by a type of vehicle. Autos, buses, trains and bicycles are examples of modes. Walking is a mode that doesn't involve a vehicle. Often, it's useful to know how many people are traveling together in an auto; therefore, driving alone and carpooling may be treated as separate modes.

organized carpool

Large employers or government agencies may often help people traveling to the same place form a carpool. The participants in an organized carpool may not have known each other before. Often, there is a formal procedure where members of an organized carpool share costs equally (for example, by taking turns driving, or by a passenger reimbursing the driver at a fixed per-mile rate).

organized vanpool

An organized vanpool is similar to an organized carpool, except more people are riding together, so a larger vehicle is used. The van may be owned by a company or government agency, instead of belonging to one of the vanpool participants.

personal business

Trips for other personal business include any errands you make for your household or your own personal benefit. Trips to the bank, drug store, dry cleaners or hardware store are examples of trips in this category.

public transportation (or public transit)

Public transportation means any mode that a member of the public can use. The vehicles used do not need to be owned by public agencies. For example, Peter Pan and Plymouth & Brockton are two private bus operators that offer commuter service. Because anyone can buy a ticket, they are considered public transportation. The MBTA, of course, is also public transportation.

route

Route means the path in the city you make as you travel. Drivers may take different routes to the same place by using different streets, or by using a freeway instead of a street. Buses are given a route number and/or name to identify the places where the bus starts and stops, and the road it travels on to get there.

telecommuting

Telecommuters use equipment such as fax machines and networked computers to work somewhere other than MIT — often at home. Because they don’t need to go in to work every day, telecommuters help reduce congestion, pollution, and of course, wear and tear on their autos.

tether account

People who live off campus and have a computer with a modem at home can dial in to use Athena. A tether account offers a communications protocol that allows users to run graphical programs such as Netscape Navigator and Eudora through Athena, rather than being restricted to a text-only screen.

C
Path Generation Algorithms (TransCAD GISDK)

This appendix contains the code to perform the path generation algorithms described in section 3.2. In general, each of these macros assumes the following types of data are available:

A node layer, containing information regarding centroids.

A link layer, containing attributes described in Tables 4-4 and 4-5, as well as temporary variables related to the link-path incidence matrix.

A turn penalty file, as described in section 4.2.3.

A route system layer containing digitized survey routes.

A skim table in which LOS variables for alternative paths are stored. In this table, each record corresponds to an individual survey respondent. The LOS variables have the form of P#var where # indicates the path (1 to 51) and var reflects the link variable being summed.

Where possible, comments on variable contents and program logic are included within the macros presented.

C.1 Link Penalty Macro

```
// ----------------------------------
// this is file://dr-doolittle.mit.edu/d|/users/sramming/tc_mac/KDPrACL2.rsc
//
// R 16 February and R 25 May 2000
// Scott Ramming
//
// Generate K different paths from each origin
// using the de la Barra heuristic
//
// initial objective: minimize congested time
//
// adaptive critical links
// -----------------------------------

Macro "Kdifft"

net_file="d:\\users\\sramming\\tc\\u90kdp4.net"
ntfh=ReadNetwork(net_file)

shared d_NetInfo

dim sel_lnks[200]
dim dir_lnks[200]
surv_dist={999999,999999,999999}

lk_view="K40 Links"

nd_view="UTPS 90 Node Data"

skm_view="KDP Resp Skim Table"

rt_view="97 Emp Survey"

views=GetViewNames()

RunMacro("TCU set network",net_file,ntfh,null)

net_hand=d_NetInfo[2][2]

d_NetInfo[3][1]=1
d_NetInfo[3][2]=0
d_NetInfo[3][3]=1

net_lk_vars=NetworkLinkVarNames(net_hand)

orig_cc_pos=ArrayPosition(net_lk_vars,{"orig_cc"},null)
dest_cc_pos=ArrayPosition(net_lk_vars,{"dest_cc"},null)
temp_imp_pos=ArrayPosition(net_lk_vars,{"temp_imp"},null)
real_rd_pos=ArrayPosition(net_lk_vars,{"real_rd"},null)

cc_pos=ArrayPosition(net_lk_vars,{"Cent_Conn"},null)

lk_flds=GetFields(lk_view,"Numeric")
```

```
d_NetInfo[4]={cc_pos,1,null}

d_NetInfo[6]={null,
              {"d:\\users\\sramming\\tc\\ftp3f4.dbf","ftp3f4.dbf","DBASE"},
              0,0,0,0}
// last four elements are network-wide turn penalties -- left, right,
// through, and U are overridden by the intersection-specific turn pen file

temp_dir ="d:\\users\\sramming\\tc\\tmp3\\"
print_dir="d:\\users\\sramming\\tc\\rts00may\\"
link_dir ="d:\\users\\sramming\\tc\\rts00may\\"

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//      User variables follow
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

orig_set="now_orig"
dest_set="now_dest"
base_name="0937"
going_home=1
// this macro is designed for one OD pair
// because it evolved from a more general macro, the origin
// and destination are expected to be in selection sets of
// a single element

use_exo=1
exo_subset=1 // 1 to specify a selection (sub)set of exog routes to use
exo_subname="survey"

printout=1
save_rts=1

max_k= 16
// number of distinct paths to generate
// constrained by the skim table
// exogenous paths must be included in this total
max_itr= 100
// maximum number of iterations to run
// each iteration may not produce a unique path
// this parameter is used to limit run times (and disk usage)

imp_inc= 0.04
// increase impedance by 4% each iteration

full=0

app_crit={22,27,32,35,38}
// ------------------------------------------------------
// this macro is really a hybrid link penalty and link
// elimination algorithm.
// at the iterations specified in app_crit, one link
// in the shortest path is removed from the network
// for the remainder of the run
// -------------------------------------------------------
num_crit=ArrayLength(app_crit)
```

```
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

// ***********************************************
// write heading of list file
// ***********************************************

if printout=1 then do
    // prn_file=OpenFile(print_dir+Word(Substring(orig_set,1,8),1)+".lst","w")
    prn_file=OpenFile(print_dir+"r"+base_name+".lst","w")
    WriteLine(prn_file,"K Different Paths with Critical Links - Version 2.0")
    WriteLine(prn_file,"Scott Ramming - May 2000")
    WriteLine(prn_file," ")
    timenow=GetDateAndTime()
    WriteLine(prn_file,"Run started at "+timenow)
    WriteLine(prn_file," ")
    WriteLine(prn_file,"Run options:")
    WriteLine(prn_file,String(max_k)+" unique paths requested")
    WriteLine(prn_file,"Run limited to "+String(max_itr)+" iterations")
    WriteLine(prn_file,"Impedance increased by "+String(100*imp_inc)+
      " percent each iteration when generating paths.")

    if num_crit=1 then WriteLine(prn_file,"Critical link removed for "+
        "unique path "+String(app_crit[1])+".")
    else if num_crit=2 then WriteLine(prn_file,"Critical link removed for "+
        "unique paths "+String(app_crit[1])+" and "+String(app_crit[2])+".")
    else if num_crit>=3 then do
        crit_txt=""
        for jl=1 to num_crit do
            if jl<num_crit then crit_txt=crit_txt+String(app_crit[jl])+", "
            else crit_txt=crit_txt+"and "+String(app_crit[num_crit])
        end
        WriteLine(prn_file,"Critical link removed for unique paths "+
                  crit_txt+".")
    end

    if use_exo=0 then WriteLine(prn_file,"No exogenous paths supplied")
    else if exo_subset=1 then WriteLine(prn_file,
             "Exogenous paths from set "+exo_subname)
         else WriteLine(prn_file,"Exogenous paths from entire "
              +rt_view+" layer")
    WriteLine(prn_file," ")
end


SetSelectInclusion("Intersecting")

// create arrays used to access link distance/fft and path size vars

l_dist={999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999}
l_fft= {999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999}

// don't initialize with l_fft=l_dist or TransCAD will map both
// arrays on to the same memory - not good!

ps_dst={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
        1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
ps_fft={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
        1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}

// Note that these arrays assume a skim table of up to 40 paths

max_ovlap={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
           0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}

nbr_end={"th","st","nd","rd","th","th","th","th","th","th"}

for kj=1 to max_k do
    if kj=1 then do
        lkdstarr={"P1_DIST"}
        lkfftarr={"P1_FFT"}
        psdstarr={{"PS1DIST",1}}
        psfftarr={{"PS1FFT",1}}
        // size of 1 is a default to change later
    end
    else do
        ins_arr={"P"+String(kj)+"_DIST"}
        lkdstarr=InsertArrayElements(lkdstarr,kj,ins_arr)
        ins_arr={"P"+String(kj)+"_FFT"}
        lkfftarr=InsertArrayElements(lkfftarr,kj,ins_arr)
        ins_arr={{"PS"+String(kj)+"DIST",1}}
        psdstarr=InsertArrayElements(psdstarr,kj,ins_arr)
        ins_arr={{"PS"+String(kj)+"FFT",1}}
        psfftarr=InsertArrayElements(psfftarr,kj,ins_arr)
    end
end

SetMap("90 UTPS Hwy Asgt")
tot_links=GetRecordCount(lk_view,null)
SetLayer("Endpoints")
tot_origs=GetSetCount(orig_set)

EnableProgressBar("Status",3)
if tot_origs>1 then CreateProgressBar("Completed Origins","False")

dest_rh=GetFirstRecord("Endpoints|"+dest_set,null)

orig_rh=GetFirstRecord("Endpoints|"+orig_set,null)

num_origs=0

while orig_rh <> null do

// ******************************************
// initialization for each origin
// ******************************************

    RunMacro("ClearLP",full)

    crit_id={"-1","-1","-1","-1","-1"}
    found_crit=0
    set_crit=0

    if printout=1 then do
        WriteLine(prn_file,"*** begin routes for origin ID "+orig_rh)
        WriteLine(prn_file,"      destination ID "+dest_rh)
        WriteLine(prn_file," ")
    end

    uniq=1 // trick to avoid multiple crit link disables

    if tot_origs>1 then UpdateProgressBar(
       "Completed Origins",Floor(num_origs/tot_origs*100))

    orgskmid=LocateRecord(skm_view+"|","OBSNO",
                        {StringToInt(base_name)},{"Exact","True"})
    // will be used later throughout this giant loop

    // select cent conns leaving from orig
    SetLayer("Endpoints")
    SetRecord(null, orig_rh)
    orig_taz=nd_view.ctps_node

    SelectNone("curr_orig")
    SelectRecord("curr_orig")

    // now do same for destination
    SetRecord(null, dest_rh)
    dest_taz=nd_view.ctps_node

    SelectNone("curr_dest")
    SelectRecord("curr_dest")

// *******************************************************
// select links to eliminate useless centroid connectors
// *******************************************************

    SetLayer("90 UTPS Hwy Asgt")
    link_id=GetFirstRecord("90 UTPS Hwy Asgt|",null)
    link_cnt=0

    CreateProgressBar(
      "Flagging Links Used for Origin and Destination","False")
    while link_id <> null do
        SetRecord(null, link_id)
        lk_view.orig_cc=0
        lk_view.dest_cc=0
        lk_view.ks_used=0
        if lk_view.anode=orig_taz or lk_view.bnode=orig_taz
          then lk_view.orig_cc=lk_view.used_cc
        if lk_view.anode=dest_taz or lk_view.bnode=dest_taz
          then lk_view.dest_cc=lk_view.mit_cc
        link_id=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
        link_cnt=link_cnt+1
        UpdateProgressBar("Flagging Links Used for Origin and Destination",
                          Floor(link_cnt/tot_links*100))
    end // link_id loop to set orig_cc
    DestroyProgressBar()

    SetView(lk_view)

    NetworkEnableDisableLinkByConditions(net_hand,null,{{"Type","Enable"}})

    UpdateNetworkCost(net_hand,"[K40 Links].ORIG_CC",null,orig_cc_pos)

    UpdateNetworkCost(net_hand,"[K40 Links].DEST_CC",null,dest_cc_pos)

    // rebuild the network disabling all cent conns except those to MIT
    // and from the current origin
    // disable all, then enable those that are real, MIT cc's or orig cc's

    NetworkEnableDisableLinkByConditions(net_hand,null,{{"Type","Disable"}})
    NetworkEnableDisableLinkByConditions(net_hand,
      {{"orig_cc","=",1},
       {"dest_cc","=",1},
       {"real_rd","=",1}},{{"Type","Enable"}})

// put all the routes leaving from the origin in a selection set

// set the exogenous/generated flag
    genflag=1
    firstgen=0
    if use_exo=0 then firstgen=1

    if use_exo=1 then do
        genflag=0
        SetLayer(rt_view)
        SelectByVicinity("orig_rts","Several","Endpoints|curr_orig",0.01)
        SelectByVicinity("dest_rts","Several","Endpoints|curr_dest",0.01)
        // eek! there may be none -- e.g., S Weymouth has mult origs

        if exo_subset>0 then do
            SetAND("orig_rts",{"orig_rts","dest_rts",exo_subname})
        end
        else do
            SetAND("orig_rts",{"orig_rts","dest_rts"})
        end

        num_exo=GetSetCount("orig_rts")

    // select the first route from the origin
        rt_id=GetFirstRecord(rt_view+"|orig_rts",null)
        if rt_id=null then do
            genflag=1
            firstgen=1
        end
        else if Substring(rt_id,1,2)="id" then do
            ck_base=Substring(rt_id,3,4)
            if base_name<>ck_base and printout=1 then
```

```
              WriteLine(prn_file,"Supplied base name ("+base_name+
                        ") disagrees with first survey route ("+
                        ck_base+").")
        end
    end // if use_exo=1

// ***********************************************
// open and write heading of link file
// ***********************************************

    if save_rts=1 then do
        lnk_file=OpenFile(link_dir+"r"+base_name+".txt","w")
        WriteLine(lnk_file,base_name+","+orig_rh+","+dest_rh+","
                  +String(max_k)+","+String(num_exo))
    end

//  set the iteration counters
    k=1
    iters=1

    CreateProgressBar("Iterations","False")

//  for each iteration
    while (k<=max_k) and (iters<=max_itr) do

        suffix=nbr_end[RealToInt(Mod(k,10)+1)]
        kq=Mod(k,100)
        if kq>10 and kq<20 then suffix="th"

        UpdateProgressBar("Iterations -- Finding the "+String(k)+
           suffix+" unique path of "+String(max_k),
           Floor(iters/max_itr*100))

//      initialization: copy cng_time to the temp_imp variable
        if iters=1 then do
            SetLayer("90 UTPS Hwy Asgt")
            link_id=GetFirstRecord("90 UTPS Hwy Asgt|",null)
            link_cnt=0
            CreateProgressBar("Initializing Link Costs","False")
            while link_id <> null do
                SetRecord(null, link_id)
                lk_view.temp_imp=lk_view.c_time
                link_id=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
                link_cnt=link_cnt+1
                UpdateProgressBar("Initializing Link Costs",
                                  Floor(link_cnt/tot_links*100))
            end // link_id loop to initialize impedances
            DestroyProgressBar()
        end // if first iteration

// ****************************************
// disable critical link
// ****************************************

        if genflag=1 & set_crit<num_crit & set_crit+1<=found_crit
          & k>=app_crit[RealToInt(Min(set_crit+1,num_crit))] & uniq=1 then do
            // uniq retains value from last loop through
            set_crit=set_crit+1
            SetRecord(null, crit_id[set_crit])
            lk_view.temp_imp=999

            if printout=1 then do
                critl_name=lk_view.name
                // may have elsewhere when selecting
                WriteLine(prn_file,"Link ID "+crit_id[set_crit]+" ("
                          +critl_name+") disabled")
                WriteLine(prn_file," ")
            end
        end

        if k=1 or genflag=1 then do
            UpdateNetworkCost(net_hand,"["+lk_view+"].temp_imp",
              null,temp_imp_pos)
        end

// ******************************
// select paths
// ******************************

// if exog/gen flag = 0 -> read from existing paths
        if genflag=0 then do
            rt_name=rt_view.Route_Name

            lnk_tmp=GetRouteLinks(rt_view,rt_name)

            nrlinks=ArrayLength(lnk_tmp)

            for ij=1 to nrlinks do
                sel_lnks[ij]=lnk_tmp[ij][1]
            end

            sel_lnk2=Subarray(sel_lnks,1,nrlinks)

        end // if genflag = 0

// else (exog/gen flag = 1 -> generate from SP routine
        else do

//      find the shortest path using temp_imp
            curr_sp=ShortestTurnPath(net_hand,RH2ID(orig_rh),RH2ID(dest_rh),
                    temp_imp_pos,
                    {{"Update Penalties",{d_NetInfo[6][3],d_NetInfo[6][4],
                                          d_NetInfo[6][5],d_NetInfo[6][6],
                                          null,d_NetInfo[6][2][1]}},
                     {"Link Type",d_NetInfo[4][1]},
                     {"Connector",d_NetInfo[4][2]}})

            nrlinks=ArrayLength(curr_sp[2])
            sel_lnk2=curr_sp[2]
        end // if genflag=1

        SelectByIDs("tmp_sp","Several",sel_lnk2)

        tmp_suffix=Substring("0123456789abcdefghijklmnopqrstuvwxyz",
            Floor(iters/32)+1,1)+
            Substring("0123456789abcdefghijklmnopqrstuvwxyz",
            RealToInt(Mod(iters,32))+1,1)

        tmp_view="r"+base_name+"i"+tmp_suffix
        tmp_fnam=temp_dir+tmp_view+".dbf"

        if k=1 then do
            skim_arr={ {"distance","sum"}, {"ff_time","sum"},
                {"c_time","sum"}, {"one","sum"},
                {"freeway","sum"},{"fwy_time","sum"},
                {"expwy","sum"}, {"expwy_time","sum"}, {"unsafe","sum"},
                {"unsf_time","sum"}, {"num_route","sum"},
                {"numrt_time","sum"}, {"toll","sum"} }
            comp_flds={"SUMDISTANC"}
            sum_arr={ "SUMDISTANC", "SUMFF_TIME", "SUMC_TIME",
                "SUMONE", "SUMFREEWAY", "SUMFWY_TIM",
                "SUMEXPWY", "SUMEXPWY_T", "SUMUNSAFE",
                "SUMUNSF_TI", "SUMNUM_ROU", "SUMNUMRT_T", "SUMTOLL" }
        end

        AggregateTable(tmp_view,lk_view+"|tmp_sp","dBASE",
          tmp_fnam,"ONE",skim_arr, null )

// ***********************
// test for uniqueness
// ***********************

        crit_iter=genflag
        if found_crit>=num_crit then crit_iter=0

        uniq=1

        tmp_rh=GetFirstRecord(tmp_view+"|",null)
        // first and only record
        comp_dsts=GetRecordValues(tmp_view,tmp_rh,comp_flds)

        if genflag=0 then surv_dist[k]=comp_dsts[1][2]
        // store distance skim of a survey (exog) route for future
        // comparisons

        if k>1 then do
            for i=1 to k-1 do
                if comp_dsts[1][2] = comp_dsts[i+1][2] then do
                // comp_dsts[1][2] is distance skim of kth path
                // comp_dsts[2...k][2] are skims of p1dist ... p<k-1>dist
                // that is, overlap distances
                // surv_dist[1..3] are distances of exog/survey routes
                    uniq=0
                    crit_iter=0
                    if printout=1 & use_exo=1 & genflag=1 & i<firstgen
                       & max_ovlap[i]<1 then do
                        WriteLine(prn_file,"Exogenous path "+String(i)+
                            " replicated at iteration "+String(iters)+
                            "; ignoring")
                        WriteLine(prn_file," ")
                        max_ovlap[i]=1
                    end // if print replication message
                end // if distances match
                else if printout=1 & use_exo=1 & genflag=1 & i<firstgen
                then if comp_dsts[i+1][2]/surv_dist[i]>max_ovlap[i]
                  then do
                    WriteLine(prn_file,"Path at iter "+String(iters)+
                      " overlaps with exogenous path "+String(i)+
                      " by about "+
                      String(Floor(comp_dsts[i+1][2]/surv_dist[i]*100))+
                      " percent.")
                    WriteLine(prn_file," ")
                    max_ovlap[i]=comp_dsts[i+1][2]/surv_dist[i]
                end // else if increasing% overlap

                if comp_dsts[i+1][2]/comp_dsts[1][2]>0.66 & genflag=1
                  & i=>firstgen then crit_iter=0

            end // for i loop
        end // if k>1

        if genflag=1 then do
            link_id=GetFirstRecord(lk_view+"|tmp_sp",null)
            sp_links=GetSetCount("tmp_sp")
            link_cnt=0
            CreateProgressBar("Increasing Link Costs","False")
            while link_id <> null do
                SetRecord(null, link_id)
                if lk_view.temp_imp<999
                    then lk_view.temp_imp=lk_view.temp_imp*(1+imp_inc)
                link_id=GetNextRecord(lk_view+"|tmp_sp",null,null)
                link_cnt=link_cnt+1
                UpdateProgressBar("Increasing Link Costs",
                                  Floor(link_cnt/sp_links*100))
            end // link_id loop to increase impedances
            DestroyProgressBar()
        end // if genflag=1

        if uniq=1 then do

// *******************************************************************
// flag the links used by this path (if not already done) in lk_view
// *******************************************************************

            link_id=GetFirstRecord(lk_view+"|tmp_sp",null)
            link_cnt=0
            while link_id <> null do
                SetRecordValues(lk_view,link_id,
                  { {"P"+String(k)+"_DIST",lk_view.distance},
                    {"P"+String(k)+"_FFT", lk_view.ff_time} })
                lk_view.ks_used=lk_view.ks_used+1
                link_id=GetNextRecord("90 UTPS Hwy Asgt|tmp_sp",null,null)
                link_cnt=link_cnt+1
            end // link_id loop to increase impedances

            temp_pos=ArrayPosition(net_lk_vars,
                       {"P"+String(k)+"_DIST"},null)
            // ShowMessage("p"+String(k)+"_dist is at position
"+String(temp_pos))

            UpdateNetworkCost(net_hand,"["+lk_view+"].p"+String(k)+"_dist",
                              null,temp_pos)

            UpdateNetworkCost(net_hand,lk_view+".p"+String(k)+"_fft",
                              null,temp_pos)

            // update dist and fft skims for path size calculation later
            tmp_id=GetFirstRecord(tmp_view+"|",null)
            l_dist[k]=tmp_view.SUMDISTANC
            l_fft[k] =tmp_view.SUMFF_TIME

// ************************************
// copy path skims to skim dataview
// ************************************

            // dumping path skims to origin-based dataview would go here
            // then can close tiny matrices earlier

            // GetRecordValues from tmp_view
            tmp_id=GetFirstRecord(tmp_view+"|",null)
            // first and only record
            skim_tmp=GetRecordValues(tmp_view,tmp_id,sum_arr)

            // SetRecordValues to skm_view
            // assemble self-path skims
            skim_put={{"p"+String(k)+"dist",skim_tmp[1][2]},
                      {"p"+String(k)+"fft" ,skim_tmp[2][2]},
                      {"p"+String(k)+"ct"  ,skim_tmp[3][2]},
                      {"p"+String(k)+"lnks",skim_tmp[4][2]},
                      {"p"+String(k)+"fwyl",skim_tmp[5][2]},
                      {"p"+String(k)+"fwyt",skim_tmp[6][2]},
                      {"p"+String(k)+"expl",skim_tmp[7][2]},
                      {"p"+String(k)+"expt",skim_tmp[8][2]},
                      {"p"+String(k)+"unsl",skim_tmp[9][2]},
                      {"p"+String(k)+"unst",skim_tmp[10][2]},
                      {"p"+String(k)+"nmrl",skim_tmp[11][2]},
                      {"p"+String(k)+"nmrt",skim_tmp[12][2]},
                      {"p"+String(k)+"toll",skim_tmp[13][2]}}
            SetRecordValues(skm_view,orgskmid,skim_put)

//          Expand the skim, comparison and sum arrays

            curskmln=ArrayLength(skim_arr)
            ins_arr={{"p"+String(k)+"_dist","sum"},
                     {"p"+String(k)+"_fft","sum"}}
            skim_arr=InsertArrayElements(skim_arr,curskmln+1,ins_arr)

            cmpfldln=ArrayLength(comp_flds)
            ins_arr={Substring("SUMP"+String(k)+"_DIST",1,10)}
            comp_flds=InsertArrayElements(comp_flds,cmpfldln+1,ins_arr)

// ***********************************
// generate and copy path directions
// ***********************************
```

```
            if printout=1 then do
                if genflag=0 then do
                    for ij=1 to nrlinks do
                    // -------------------------------
                    // in a route system,  dir 1 = w/topo, -1 = against topo
                    // in a shortest path, dir 0 = w/topo,  1 = against topo
                    // -------------------------------
                        dir_lnks[ij]=Floor(0.5-(lnk_tmp[ij][1]/2))
                    end
                    dir_lnk2=Subarray(dir_lnks,1,nrlinks)
                end
                else do
                    dir_lnk2=curr_sp[3]
                end

                tmp_dnam=temp_dir+tmp_view+".txt"

                PathDirections(sel_lnk2,dir_lnk2,lk_view+".Name",
                    tmp_dnam,lk_view+".ID")

                // now copy the whole thing over to the big file

                WriteLine(prn_file,"Unique path "+String(k)+" ("+
                    Word("exogenous generated",genflag+1)+"; iter "+
                    String(iters)+") :")

                if genflag=0 then WriteLine(prn_file,"(based on "+
                    rt_name+".)")
                else WriteLine(prn_file,"(total impedance = "+
                    String(curr_sp[1])+".)")

                dir_file=OpenFile(tmp_dnam,"r")
                while !FileAtEOF(dir_file) do
                   tmp_line=ReadLine(dir_file)
                   if StringLength(tmp_line)>0 then
                       WriteLine(prn_file,tmp_line)
               end
               CloseFile(dir_file)
               WriteLine(prn_file," ")

            end // if printout=1

// *************************************
// select critical link for disabling
// *************************************

            if genflag=1 & crit_iter=1 then do
                SetLayer("90 UTPS Hwy Asgt") // needed?
                critl_id=GetFirstRecord(lk_view+"|tmp_sp",
                    {{"Imp_Vol","Descending"}})
                SetRecord(null, critl_id)
                critl_name=lk_view.name
                if StringLength(critl_name)=0 then critl_name="unnamed"

                // check that the crit link hasn't been used before
                crit_uniq=1
                for ll=1 to found_crit do
                    if crit_id[ll]=critl_id then crit_uniq=0
                end
                if crit_uniq=1 then do
                    found_crit=found_crit+1
                    crit_id[found_crit]=critl_id

                    if printout=1 then do
                        WriteLine(prn_file,"Link ID "+critl_id
                                  +" ("+critl_name
                                  +") selected as critical")
                        WriteLine(prn_file," ")
                    end
                end // if proposed critical link was unique

            end // if generating paths and this is a critical iter

// ************************************
// write link IDs to link file
// ************************************

            if save_rts=1 then do
                WriteLine(lnk_file,String(k)+","+String(1-genflag)+","+
                          String(nrlinks))
                for rr=1 to nrlinks do
                    WriteLine(lnk_file,String(sel_lnk2[rr]))
                end
            end

        end // if uniq=1

        CloseView(tmp_view)

        SelectNone("tmp_sp")

        k=k+uniq
        iters=iters+1

//      if still reading exog paths
        if genflag=0 then do
            // get next route
            rt_id=GetNextRecord(rt_view+"|orig_rts",null,null)
            if rt_id=null then do
                genflag=1
                firstgen=k
            end
        end // if genflag was 0

    end // do over iters aka while k<=max_k

    DestroyProgressBar()  // for iters

    did_k=max_k
    if (iters>max_itr)and(uniq=0) then do
        did_k=k-1
        if printout=1 then do
            WriteLine(prn_file,"Stopped after "+String(did_k)+
               " unique paths because maximum iterations reached")
            WriteLine(prn_file," ")
        end
    end

    if did_k<max_k and save_rts=1 then do
        for kr=did_k+1 to max_k do
           WriteLine(lnk_file,String(kr)+",0,0")
           // no links in this non-generated path
        end
    end

    if save_rts=1 then CloseFile(lnk_file)

    // *********************************************
    // now calculate and store path size variables
    // *********************************************
    // initialize
    for ij=1 to did_k do
        ps_dst[ij]=0
        ps_fft[ij]=0
    end

    // loop over links
    link_id=GetFirstRecord(lk_view+"|",null)
    link_cnt=0
    CreateProgressBar("Calculating Path Sizes","False")

    oldstarr=Subarray(lkdstarr,1,did_k)
    olfftarr=Subarray(lkfftarr,1,did_k)

    while link_id <> null do
        // get link dist/fft flag arrays

        lk_dsts=GetRecordValues(lk_view,link_id,oldstarr)
        lk_ffts=GetRecordValues(lk_view,link_id,olfftarr)

        // loop over paths to find shortest ones (or if no paths use link)
        path_cnt=0
        firstpth=0
        for jj=1 to did_k do
            if lk_dsts[jj][2]>0 then do
                if path_cnt=0 then do
                    min_dist=l_dist[jj]
                    min_fft =l_fft[jj]
                    firstpth=jj
                end // if path_cnt=0
                else do
                    min_dist=Min(min_dist,l_dist[jj])
                    min_fft =Min(min_fft ,l_fft[jj])
                end
                path_cnt=path_cnt+1
            end // if link used by path
        end // jj loop to search for min dist and fft paths

        // now there are three outcomes
```

```
        // path_cnt=0 - no paths use this link - go on to next
        // path_cnt=1 - link is unique to path - size is eazy
        // path_cnt>1 - use existing code to calculate size

        if path_cnt>0 then do
            if lk_view.cent_conn>0 then do // centroid connectors
                for jj=firstpth to did_k do
                    if lk_dsts[jj][2]>0 then do
                    // link is used for path jj
                        ps_dst[jj]=ps_dst[jj]+lk_dsts[jj][2]
                        ps_fft[jj]=ps_fft[jj]+lk_ffts[jj][2]
                    end
                end // for jj
            end // if centroid connector
            else do // normal links
                if path_cnt=1 then do
                    ps_dst[firstpth]=ps_dst[firstpth]+lk_dsts[firstpth][2]
                    ps_fft[firstpth]=ps_fft[firstpth]+lk_ffts[firstpth][2]
                end
                else do // path_cnt > 1
                    // loop over paths to sum up denominator
                    dstdenom=0
                    fftdenom=0
                    for jj=1 to did_k do
                        if lk_dsts[jj][2]>0 then do
                        // link is used for path jj
                            dstdenom=dstdenom+min_dist/l_dist[jj]
                            fftdenom=fftdenom+min_fft/l_fft[jj]
                        end
                    end // for jj loop to sum denominator
                    // loop over paths to add arc-specific parts
                    // of path size sum
                    for jj=1 to did_k do
                        if lk_dsts[jj][2]>0 then do

                        // changed so that link size contributions sum to 1

                            ps_dst[jj]=ps_dst[jj]+lk_dsts[jj][2]*
                                       (min_dist/l_dist[jj])/dstdenom
                            ps_fft[jj]=ps_fft[jj]+lk_ffts[jj][2]*
                                       (min_fft / l_fft[jj])/fftdenom
                        end // if link used for path jj
                    end // for jj loop
                end // else path_cnt>1 block
            end // if ... else regular link
        end // if path_cnt>0 block

        link_id=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
        link_cnt=link_cnt+1
        UpdateProgressBar("Calculating Path Sizes",
                           Floor(link_cnt/tot_links*100))
    end // loop over links
    DestroyProgressBar()

    // now finish the path size calculation by normalizing by the constants
    // and put the path sizes in an array to send to the skim table
    for jj=1 to did_k do
```

```
        ps_dst[jj]=ps_dst[jj]/(l_dist[jj])
        psdstarr[jj][2]=ps_dst[jj]

        ps_fft[jj]=ps_fft[jj]/(l_fft[jj])
        psfftarr[jj][2]=ps_fft[jj]
    end // calculate and prepare path size loop

    // put path sizes in the skim table

    opdstarr=Subarray(psdstarr,1,did_k)
    opfftarr=Subarray(psfftarr,1,did_k)

    SetRecordValues(skm_view,orgskmid,opdstarr)
    SetRecordValues(skm_view,orgskmid,opfftarr)

    SetRecordValues(skm_view,orgskmid,{{"PS_Def",1}})

    SetMap("90 UTPS Hwy Asgt")
    SetLayer("Endpoints")

    num_origs=num_origs+1
    if num_origs<tot_origs then
      orig_rh=GetNextRecord("Endpoints|"+orig_set,null,null)
    else orig_rh=null
end // do over origins
if tot_origs>1 then DestroyProgressBar() // for origins
DisableProgressBar()

// reset these to the Caliper defaults
EnableProgressBar("Status ...",1)
DisableProgressBar()

// ************************************
// close off the print file
// ************************************

if printout=1 then do
    timenow=GetDateAndTime()
    WriteLine(prn_file,"Run finished at "+timenow)
    CloseFile(prn_file)
end


if going_home=1 then
ShowMessage("K Different Paths Macro finished successfully!"+
            "\nIf you need to use Dr. Doolittle, just close TransCAD."+
            "\nThere's no need to save the files; however, you may want"+
            "\nto reboot to clear up memory.")
else
ShowMessage("K Different Paths Macro finished successfully!"+
            "\nScott will start a new run shortly.")

endMacro

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Macro "ClearLP" (full)
```

```
    lk_view="K40 Links"
    lk_layr="90 UTPS Hwy Asgt"

    max_k=40

    for kj=1 to max_k do
        if kj=1 then do
            lpdstarr={{"P1_DIST",0}}
            lpfftarr={{"P1_FFT" ,0}}
        end
        else do
            ins_arr={{"P"+String(kj)+"_DIST",0}}
            lpdstarr=InsertArrayElements(lpdstarr,kj,ins_arr)
            ins_arr={{"P"+String(kj)+"_FFT" ,0}}
            lpfftarr=InsertArrayElements(lpfftarr,kj,ins_arr)
        end
    end

    SetLayer("90 UTPS Hwy Asgt")

    tot_links=GetRecordCount(lk_view,null)

    link_rh=GetFirstRecord("90 UTPS Hwy Asgt|",null)
    link_cnt=0
    last_pct=0

    CreateProgressBar("Clearing Link Flags","False")
    while (link_rh <> null) and (link_cnt<tot_links) do
        SetRecord(null, link_rh)

        val_temp=GetRecordValues(lk_view,link_rh,{"KS_USED"})
        val_ks_used=val_temp[1][2]

        if (full>0) or (val_ks_used>0) then do
            SetRecordValues(lk_view,link_rh,lpdstarr)
            SetRecordValues(lk_view,link_rh,lpfftarr)
            SetRecordValues(lk_view,link_rh,{{"KS_USED",0}})
        end

        link_rh=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
        link_cnt=link_cnt+1
        if Floor(link_cnt/tot_links*100)>last_pct then do
            UpdateProgressBar("Cleared "+String(link_cnt)+" Link Flags",
                              Floor(link_cnt/tot_links*100))
            last_pct=Floor(link_cnt/tot_links*100)
        end
    end // link_rh loop
    DestroyProgressBar()

EndMacro // ClearLP
```

C.2 Link Elimination Macro

```
// ------------------------------------
// this is file://dr-doolittle.mit.edu/d|/users/sramming/tc_mac/KDP_lm1.rsc
//
```

```
// M 7 August 2000
// Scott Ramming
//
// Generate K different paths from each origin
// using the link elimination (DynaMIT) heuristic
//
// initial objective: minimize congested time
// ------------------------------------

Macro "KdifftLM"

net_file="d:\\users\\sramming\\tc\\u90kdp4.net"
ntfh=ReadNetwork(net_file)
shared d_NetInfo

dim sel_lnks[200] // see max_itr below
dim dir_lnks[200]
dim fsp_lnks[200]
surv_dist={999999,999999,999999}

lk_view="K40 Links"
lk_layr="90 UTPS Hwy Asgt"

nd_view="UTPS 90 Node Data"
nd_layr="Endpoints"

skm_view="KDP Resp Skim Table"

rt_view="97 Emp Survey"
rt_layr="97 Emp Survey"

views=GetViewNames()

RunMacro("TCU set network",net_file,ntfh,null)

net_hand=d_NetInfo[2][2]

d_NetInfo[3][1]=1
d_NetInfo[3][2]=0
d_NetInfo[3][3]=1

net_lk_vars=NetworkLinkVarNames(net_hand)

orig_cc_pos=ArrayPosition(net_lk_vars,{"orig_cc"},null)
dest_cc_pos=ArrayPosition(net_lk_vars,{"dest_cc"},null)
real_rd_pos=ArrayPosition(net_lk_vars,{"real_rd"},null)

cc_pos=ArrayPosition(net_lk_vars,{"Cent_Conn"},null)

lk_flds=GetFields(lk_view,"Numeric")

d_NetInfo[4]={cc_pos,1,null}

d_NetInfo[6]={null,
              {"d:\\users\\sramming\\tc\\ftp3f4.dbf","ftp3f4.dbf","DBASE"},
              0,0,0,0}
// last four elements are network-wide turn penalties --
// left, right, through, U -- are
// overridden by the intersection-specific turn pen file

temp_dir ="d:\\users\\sramming\\tc\\tmp3\\"
print_dir="d:\\users\\sramming\\tc\\lm00aug\\"
link_dir ="d:\\users\\sramming\\tc\\lm00aug\\"

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
//      User variables follow
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

going_home=0

exo_subname="id317"

printsumm=1
printdirs=1
save_rts=1
calc_ps=0 // don't calculate path sizes for now

max_k= 40
// this is the limit to the number of paths in the skim view
// real number of paths determined by links on the shortest path
// and network topology

full=0

imp_var="distance" // "c_time"
// link variable to minimize when producing paths

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

imp_pos=ArrayPosition(net_lk_vars,{imp_var},null)

if imp_pos=0 then do
    err_lev=4
    ShowMessage("Label "+imp_var+" was not found in the network.\n"
              +"using C_Time instead.")
    imp_pos=ArrayPosition(net_lk_vars,{"c_time"},null)
end

SetSelectInclusion("Intersecting")

// create arrays used to access link distance/fft and path size vars

l_dist={999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999}
l_fft= {999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999,
        999999,999999,999999,999999,999999,999999,999999,999999}

ps_dst={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
        1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
ps_fft={1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,
        1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1}
// since for now, my skim table only has room for K=40 paths

nbr_end={"th","st","nd","rd","th","th","th","th","th","th"}

// create variable list for link/path incidence matrix refs
for kj=1 to max_k do
    if kj=1 then do
        lkdstarr={"P1_DIST"}
        lkfftarr={"P1_FFT"}
        psdstarr={{"PS1DIST",1}}
        psfftarr={{"PS1FFT",1}}
        // size of 1 is a default to change later
    end
    else do
        ins_arr={"P"+String(kj)+"_DIST"}
        lkdstarr=InsertArrayElements(lkdstarr,kj,ins_arr)
        ins_arr={"P"+String(kj)+"_FFT"}
        lkfftarr=InsertArrayElements(lkfftarr,kj,ins_arr)
        ins_arr={{"PS"+String(kj)+"DIST",1}}
        psdstarr=InsertArrayElements(psdstarr,kj,ins_arr)
        ins_arr={{"PS"+String(kj)+"FFT",1}}
        psfftarr=InsertArrayElements(psfftarr,kj,ins_arr)
    end
end

// open summary file of
// obsno,best ovl,k found,eliminated link

if printsumm=1 then do

summ_file=OpenFile(print_dir+"lm"+Substring(exo_subname,1,6)+".lst","w")
        WriteLine(summ_file,
          "K Different Paths - Link Elimination - Route-Based Version 1.0")
        WriteLine(summ_file,"Scott Ramming - December 2000")
        WriteLine(summ_file,"")
        timenow=GetDateAndTime()
        WriteLine(summ_file,"Run started at "+timenow)
        WriteLine(summ_file,"")
        WriteLine(summ_file,"Run options:")
        WriteLine(summ_file,"Skim view allows up to "+String(max_k)
          +" unique paths")
        WriteLine(summ_file,"Path size calculations "
            +Substring("not requested. . . .",1+calc_ps*4,13-calc_ps*4))

        WriteLine(summ_file,"Exogenous paths from set "+exo_subname)
        WriteLine(summ_file,"")
        WriteLine(summ_file,"obsno,best ovl,best k,elim link")
end

SetMap("90 UTPS Hwy Asgt")
SetLayer(lk_layr)
```

```
tot_links=GetRecordCount(lk_view,null)
SetLayer(rt_layr)
tot_rts=GetRecordCount(rt_layr,exo_subname)

EnableProgressBar("Status",3)
if tot_rts>1 then CreateProgressBar("Completed Routes","False")

rt_rh=GetFirstRecord(rt_layr+"|"+exo_subname,null)

did_rts=0

while (rt_rh <> null) and (did_rts<tot_rts) do

// ******************************************
// initialization for each route
// ******************************************

    max_ovlap={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
    best_k   ={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
               0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
    best_rem_lk=-1

    SetLayer(rt_layr)
    SetRecord(null, rt_rh)
    SelectNone("mac_temp")
    SelectRecord("mac_temp")

    rt_name=rt_view.Route_Name
    base_name=Substring(rt_name+"xxsq"+String(did_rts),3,4)

    // ***********************************************
    // write heading of list file
    // ***********************************************

    if printdirs=1 then do
        prn_file=OpenFile(print_dir+"r"+base_name+".lst","w")
        WriteLine(prn_file,
          "K Different Paths - Link Elimination - Route-Based Version 1.0")
        WriteLine(prn_file,"Scott Ramming - December 2000")
        WriteLine(prn_file," ")
        timenow=GetDateAndTime()
        WriteLine(prn_file,"Run started at "+timenow)
        WriteLine(prn_file," ")
        WriteLine(prn_file,"Run options:")
        WriteLine(prn_file,"Skim view allows up to "+String(max_k)
          +" unique paths")
        WriteLine(prn_file,"Path size calculations "
            +Substring("not requested. . . .",1+calc_ps*4,13-calc_ps*4))

        WriteLine(prn_file,"Exogenous paths from set "+exo_subname)
        WriteLine(prn_file," ")
    end

    SetLayer(lk_layr)
    RunMacro("ClearLP",full)
```

```
exempt_orig=0
exempt_dest=0
// number of additional links not to eliminate
// so network isn't disconnected
// use exempt_dest=1 for Mem Dr at Burton-Conner

// get origin and destination IDs

lnk_tmp=GetRouteLinks(rt_view,rt_name)

nrlinks=ArrayLength(lnk_tmp)
occ_id=lnk_tmp[1][1]
dcc_id=lnk_tmp[nrlinks][1]

SetLayer(lk_layr)
occ_end=GetEndpoints(occ_id)
dcc_end=GetEndpoints(dcc_id)

SetLayer(nd_layr)
occa_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{occ_end[1]},
                         {"Exact","True"})
SetRecord(nd_layr,occa_rh)
if nd_view.CENTROID>0 then do
    orig_id=occ_end[1]
    SelectNone("curr_orig")
    SelectRecord("curr_orig") // for debugging only
end
else do
    orig_id=occ_end[2]
    occb_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{occ_end[2]},
                         {"Exact","True"})
    SetRecord(null,occb_rh)
    SelectNone("curr_orig")
    SelectRecord("curr_orig")
end

dccb_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{dcc_end[2]},
                     {"Exact","True"})
SetRecord(nd_layr,dccb_rh) // since most dest are new
if nd_view.CENTROID>0 then do
    dest_id=dcc_end[2]
    SelectNone("curr_dest")
    SelectRecord("curr_dest")
end
else do
    dest_id=dcc_end[1]
    dcca_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{dcc_end[1]},
                         {"Exact","True"})
    SetRecord(null,dcca_rh)
    SelectNone("curr_dest")
    SelectRecord("curr_dest")
end

if ArrayPosition({509,512,1059,2658},{orig_id},null)>0 then exempt_orig=1
if dest_id=13002 then exempt_dest=1

if printdirs=1 then do
```

```
        WriteLine(prn_file,"*** routes for id"+base_name+":")
        WriteLine(prn_file,"      origin ID "+String(orig_id))
        WriteLine(prn_file,"      destination ID "+String(dest_id))
        WriteLine(prn_file," ")
    end

    uniq=1 // test if route has been found before

    if tot_rts>1 then
      UpdateProgressBar("Completed "+String(did_rts)
        +" routes",Floor(did_rts/tot_rts*100))

    orgskmid=LocateRecord(skm_view+"|","OBSNO",
                          {StringToInt(base_name)},{"Exact","True"})
    // will be used later throughout this giant loop

    SetLayer("90 UTPS Hwy Asgt")

    SelectNone("elim_lk")

    link_id=GetFirstRecord("90 UTPS Hwy Asgt|",null)
    link_cnt=0

    SetView(lk_view)

    NetworkEnableDisableLinkByConditions(net_hand,null,{{"Type","Enable"}})

// set the exogenous/generated flag
    genflag=0
    firstgen=0

// ***********************************************
// open and write heading of link file
// ***********************************************

    if save_rts=1 then do
        lnk_file=OpenFile(link_dir+"r"+base_name+".txt","w")
        WriteLine(lnk_file,base_name+","+orig_rh+","+dest_rh+","
                   +String(max_k)+","+String(num_exo))
    end

//  set the iteration counters
    k=1
    iters=1
    max_itr=200
    // because that's how large I dimensioned the arrays above

    CreateProgressBar("Iterations","False")

    finished=0
    found_sp=0
    rem_sp_lk=0
    // use this to keep track of where we are on the links in the SP
    elim_id=-1
    elim_name="none removed"

//  for each iteration
```

```
    while (k<=max_k) and (finished=0) do

        suffix=nbr_end[RealToInt(Mod(k,10)+1)]
        kq=Mod(k,100)
        if kq>10 and kq<20 then suffix="th"

        if genflag=0 then do // using exogpaths
            UpdateProgressBar("Iterations -- Finding the "+String(k)+
               suffix+" unique exogenous path.",
               Floor(iters/max_itr*100))
        end
        else if found_sp=0 then do // now searching for it
            UpdateProgressBar("Iterations -- Finding the shortest path "
               +"and the "+String(k)+suffix+" unique path.",
               Floor(iters/max_itr*100))
        end
        else do // link elimination paths
            UpdateProgressBar("Iterations -- Finding the "+String(k)+
               suffix+" unique path of up to "+String(max_k),
               Floor(rem_sp_lk/nremlinks*100))
        end

        SetLayer(lk_layr)

// this big loop does the following:
// 1. skim all of the exogenous paths (usually 1 or 2)
// 2. build the least c_time path
// 3. build new paths by removing 1 link found in step 2

// ***********************************************
// disable the eliminated link if we're at stage 3
// ***********************************************

        if genflag=1 & found_sp=1 then do
            rem_sp_lk=rem_sp_lk+1
            elim_id=sp_links[rem_sp_lk]
            elim_rh=LocateRecord(lk_view+"|","ID",
                         {elim_id},{"Exact","True"})
            SetRecord(null,elim_rh)
            elim_name=lk_view.NAME
            SelectNone("elim_lk")
            SelectByIDs("elim_lk","Several",{elim_id})
            ChangeLinkStatus(net_hand,lk_view+"|elim_lk",
                { {"Link ID","["+lk_layr+"].ID"} ,
                  {"Type", "Disable"} } )
        end

// *******************************
// select paths
// *******************************

// if exog/gen flag = 0 -> read from existing paths
        if genflag=0 then do

            for ij=1 to nrlinks do
                sel_lnks[ij]=lnk_tmp[ij][1]
            end
            sel_lnk2=Subarray(sel_lnks,1,nrlinks)
        end // if genflag = 0

// else (exog/gen flag = 1 -> generate from SP routine
        else do

//      find the shortest path
            curr_sp=ShortestTurnPath(net_hand,orig_id,dest_id,
                    imp_pos,
                    {{"Update Penalties",{d_NetInfo[6][3],d_NetInfo[6][4],
                                          d_NetInfo[6][5],d_NetInfo[6][6],
                                          null,d_NetInfo[6][2][1]}},
                     {"Link Type",d_NetInfo[4][1]},
                     {"Connector",d_NetInfo[4][2]}})

            nrlinks=ArrayLength(curr_sp[2])
            sel_lnk2=curr_sp[2]

            if found_sp=0 then do // this must be the first generated path
                found_sp=1

                // check for U-turns
                if sel_lnk2[2+exempt_orig]=sel_lnk2[3+exempt_orig]
                  then exempt_orig=exempt_orig+1
                if sel_lnk2[nrlinks-1-exempt_dest]=
                  sel_lnk2[nrlinks-2-exempt_dest]
                  then exempt_dest=exempt_dest+1

                nremlinks=nrlinks-4-exempt_orig-exempt_dest

                if nremlinks<1 then do
                // no links to eliminate
                    ShowMessage("Number of exluded/exempt links ("
                      +String(4+exempt_orig+exempt_dest)
                      +") exceeds the number of links in the shortest path ("
                      +String(nrlinks)+").")
                    if printdirs=1 then
                      WriteLine(prn_file,"Number of exluded/exempt links ("
                      +String(4+exempt_orig+exempt_dest)
                      +") exceeds the number of links in the shortest path ("
                      +String(nrlinks)+").")
                    finished=1
                end
                else do

                    sp_links=Subarray(sel_lnk2,3+exempt_orig,nremlinks)
                    // don't bother removing the first or last two links
                    // (one CC and one real) because paths are constrained
                    // to use them

                end
            end

        end // if genflag=1

        SelectByIDs("tmp_sp","Several",sel_lnk2)
        tmp_suffix=Substring("0123456789abcdefghijklmnopqrstuvwxyz",
            Floor(iters/32)+1,1)+
            Substring("0123456789abcdefghijklmnopqrstuvwxyz",
            RealToInt(Mod(iters,32))+1,1)

        tmp_view="r"+base_name+"i"+tmp_suffix
        tmp_fnam=temp_dir+tmp_view+".dbf"

        if k=1 then do
            skim_arr={ {"distance","sum"}, {"ff_time","sum"},
                {"c_time","sum"}, {"one","sum"},
                {"freeway","sum"},{"fwy_time","sum"},
                {"expwy","sum"}, {"expwy_time","sum"}, {"unsafe","sum"},
                {"unsf_time","sum"}, {"num_route","sum"},
                {"numrt_time","sum"}, {"toll","sum"} }
            comp_flds={"SUMDISTANC"}
            sum_arr={ "SUMDISTANC", "SUMFF_TIME", "SUMC_TIME",
                "SUMONE", "SUMFREEWAY", "SUMFWY_TIM",
                "SUMEXPWY", "SUMEXPWY_T", "SUMUNSAFE",
                "SUMUNSF_TI", "SUMNUM_ROU", "SUMNUMRT_T", "SUMTOLL" }
        end

        AggregateTable(tmp_view,lk_view+"|tmp_sp","dBASE",
          tmp_fnam,"ONE",skim_arr, null )

// ***********************
// test for uniqueness
// ***********************

        uniq=1

        tmp_rh=GetFirstRecord(tmp_view+"|",null)
        // first and only record
        comp_dsts=GetRecordValues(tmp_view,tmp_rh,comp_flds)

        if genflag=0 then surv_dist[k]=comp_dsts[1][2]
        // store distance skim of a survey (exog) route for future
        // comparisons

        if k>1 then do
            for i=1 to k-1 do
                if comp_dsts[1][2] = comp_dsts[i+1][2] then do
                // comp_dsts[1][2] is distance skim of kth path
                // comp_dsts[2...k][2] are skims of p1dist ... p<k-1>dist
                // that is, overlap distances
                // surv_dist[1..3] are distances of exog/survey routes
                    uniq=0
                    if printdirs=1 & genflag=1 & i<firstgen
                       & max_ovlap[i]<1 then do
                        WriteLine(prn_file,"Exogenous path "+String(i)+
                            " replicated at iteration "+String(iters)+
                            "; ignoring")

                        if genflag=1 and rem_sp_lk>0 then WriteLine(prn_file,
                          "Found by removing link "+String(elim_id)+" ("
                          +elim_name+").")
                        WriteLine(prn_file," ")
                        max_ovlap[i]=1
                        best_k[i]=k-0.5
                        if elim_id>0 then best_rem_lk=elim_id
                    end // if print replication message
                end // if distances match
                else if printdirs=1 & genflag=1 & i<firstgen
                // need then if for when i>=firstgen
                then if comp_dsts[i+1][2]/surv_dist[i]>max_ovlap[i]
                  then do
                    WriteLine(prn_file,"Path at iter "+String(iters)+
                      " overlaps with exogenous path "+String(i)+
                      " by about "+
                      String(Floor(comp_dsts[i+1][2]/surv_dist[i]*100))+
                      " percent.")
                    WriteLine(prn_file," ")
                    max_ovlap[i]=comp_dsts[i+1][2]/surv_dist[i]
                    best_k[i]=k
                    if elim_id>0 then best_rem_lk=elim_id
                end // else if increasing% overlap
            end // for i loop
        end // if k>1

        if uniq=1 then do

// *****************************************************************
// flag the links used by this path (if not already done) in lk_view
// *****************************************************************

            link_id=GetFirstRecord(lk_view+"|tmp_sp",null)
            link_cnt=0
            while link_id <> null do
                SetRecordValues(lk_view,link_id,
                  { {"P"+String(k)+"_DIST",lk_view.distance},
                    {"P"+String(k)+"_FFT", lk_view.ff_time} })
                lk_view.ks_used=lk_view.ks_used+1
                link_id=GetNextRecord("90 UTPS Hwy Asgt|tmp_sp",null,null)
                link_cnt=link_cnt+1
            end // link_id loop to increase impedances

            // update dist and fft skims for path size calculation later
            tmp_id=GetFirstRecord(tmp_view+"|",null)
            l_dist[k]=tmp_view.SUMDISTANC
            l_fft[k] =tmp_view.SUMFF_TIME

// *************************************
// copy path skims to skim dataview
// *************************************

            // GetRecordValues from tmp_view
            tmp_id=GetFirstRecord(tmp_view+"|",null)
            // first and only record
            skim_tmp=GetRecordValues(tmp_view,tmp_id,sum_arr)

            // SetRecordValues to skm_view
            // assemble self-path skims
            skim_put={{"p"+String(k)+"dist",skim_tmp[1][2]},
                      {"p"+String(k)+"fft" ,skim_tmp[2][2]},
                      {"p"+String(k)+"ct"  ,skim_tmp[3][2]},
                      {"p"+String(k)+"lnks",skim_tmp[4][2]},
                      {"p"+String(k)+"fwyl",skim_tmp[5][2]},
                      {"p"+String(k)+"fwyt",skim_tmp[6][2]},
                      {"p"+String(k)+"expl",skim_tmp[7][2]},
                      {"p"+String(k)+"expt",skim_tmp[8][2]},
                      {"p"+String(k)+"unsl",skim_tmp[9][2]},
                      {"p"+String(k)+"unst",skim_tmp[10][2]},
                      {"p"+String(k)+"nmrl",skim_tmp[11][2]},
                      {"p"+String(k)+"nmrt",skim_tmp[12][2]},
                      {"p"+String(k)+"toll",skim_tmp[13][2]}}

            SetRecordValues(skm_view,orgskmid,skim_put)

//          Expand the skim, comparison and sum arrays

            curskmln=ArrayLength(skim_arr)
            ins_arr={{"p"+String(k)+"_dist","sum"},
                     {"p"+String(k)+"_fft","sum"}}
            skim_arr=InsertArrayElements(skim_arr,curskmln+1,ins_arr)

            cmpfldln=ArrayLength(comp_flds)
            // a null counts as one element
            ins_arr={Substring("SUMP"+String(k)+"_DIST",1,10)}
            comp_flds=InsertArrayElements(comp_flds,cmpfldln+1,ins_arr)

// *************************************
// generate and copy path directions
// *************************************

            if printdirs=1 then do
                if genflag=0 then do
                    for ij=1 to nrlinks do
                        dir_lnks[ij]=lnk_tmp[ij][1]
                    end
                    dir_lnk2=Subarray(dir_lnks,1,nrlinks)
                end
                else do
                    dir_lnk2=curr_sp[3]
                end

                tmp_dnam=temp_dir+tmp_view+".txt"

                PathDirections(sel_lnk2,dir_lnk2,lk_view+".Name",
                    tmp_dnam,lk_view+".ID")

                // now copy the whole thing over to the big file

                WriteLine(prn_file,"Unique path "+String(k)+" ("+
                    Word("exogenous generated",genflag+1)+"; iter "+
                    String(iters)+") :")

                if genflag=0 then WriteLine(prn_file,"(based on "+
                    rt_name+".)")
                else WriteLine(prn_file,"(total impedance = "+
                    String(curr_sp[1])+".)")

                if genflag=1 and rem_sp_lk>0 then WriteLine(prn_file,
                    "Found by removing link "+String(elim_id)+" ("
                    +elim_name+").")

                dir_file=OpenFile(tmp_dnam,"r")
                while !FileAtEOF(dir_file) do
                   tmp_line=ReadLine(dir_file)
                   if StringLength(tmp_line)>0 then
                       WriteLine(prn_file,tmp_line)
               end
               CloseFile(dir_file)
               WriteLine(prn_file," ")

            end // if printout=1

// *************************************
// write link IDs to link file
// *************************************

            if save_rts=1 then do
                WriteLine(lnk_file,String(k)+","+String(1-genflag)+","+
                          String(nrlinks))
                for rr=1 to nrlinks do
                    WriteLine(lnk_file,String(sel_lnk2[rr]))
                end
            end

        end // if uniq=1

// *************************************
// restore any eliminated/disabled link
// *************************************

        SetLayer(lk_layr) // needed?

        if genflag=1 & found_sp=1 & rem_sp_lk>0 then do
            ChangeLinkStatus(net_hand,lk_view+"|elim_lk",
                { {"Link ID","["+lk_layr+"].ID"} ,
                  {"Type", "Enable"} } )
            if rem_sp_lk=>nremlinks then finished=1
            // rem_sp_lk should never be > nremlinks
        end // reenabling links

        CloseView(tmp_view)

        SelectNone("tmp_sp")

        k=k+uniq
        iters=iters+1
        if genflag=0 then do
            genflag=1 // since macro now based on one rt = OD
            firstgen=2
        end

    end // do over iters aka while k<=max_k
```

```
DestroyProgressBar()  // for iters

if finished=1 then did_k=did_k-1
// because it didn't find that kth path it was looking for
// only affects path size calculation
if (iters>max_itr)and(uniq=0) then do
    did_k=k-1
    if printout=1 then do
        WriteLine(prn_file,"Stopped after "+String(did_k)+
           " unique paths because maximum iterations reached")
        WriteLine(prn_file," ")
    end
end

if did_k<max_k and save_rts=1 then do
    for kr=did_k+1 to max_k do
       WriteLine(lnk_file,String(kr)+",0,0")
       // no links in this non-generated path
    end
end
if save_rts=1 then CloseFile(lnk_file)

// ********************************************
// now calculate and store path size variables
// ********************************************
// initialize

if calc_ps>0 then do // calc path size branch

    for ij=1 to did_k do
        ps_dst[ij]=0
        ps_fft[ij]=0
    end

    // loop over links
    link_id=GetFirstRecord(lk_view+"|",null)
    link_cnt=0
    CreateProgressBar("Calculating Path Sizes","False")

    oldstarr=Subarray(lkdstarr,1,did_k)
    olfftarr=Subarray(lkfftarr,1,did_k)

    while link_id <> null do
        // get link dist/fft flag arrays

        lk_dsts=GetRecordValues(lk_view,link_id,oldstarr)
        lk_ffts=GetRecordValues(lk_view,link_id,olfftarr)

        // loop over paths to find shortest ones
        // (or if no paths use link)
        path_cnt=0
        firstpth=0
        for jj=1 to did_k do
            if lk_dsts[jj][2]>0 then do
                if path_cnt=0 then do
                    min_dist=l_dist[jj]
                    min_fft =l_fft[jj]
                    firstpth=jj
                end // if path_cnt=0
                else do
                    min_dist=Min(min_dist,l_dist[jj])
                    min_fft =Min(min_fft ,l_fft[jj])
                end
                path_cnt=path_cnt+1
            end // if link used by path
        end // jj loop to search for min dist and fft paths

        // now there are three outcomes
        // path_cnt=0 - no paths use this link - go on to next
        // path_cnt=1 - link is unique to path - size is eazy
        // path_cnt>1 - use existing code to calculate size

        if path_cnt>0 then do
            if lk_view.cent_conn>0 then do // centroid connectors
                for jj=firstpth to did_k do
                    if lk_dsts[jj][2]>0 then do
                    // link is used for path jj
                        ps_dst[jj]=ps_dst[jj]+lk_dsts[jj][2]
                        ps_fft[jj]=ps_fft[jj]+lk_ffts[jj][2]
                    end
                end // for jj
            end // if centroid connector
            else do // normal links
                if path_cnt=1 then do
                    ps_dst[firstpth]=ps_dst[firstpth]
                      +lk_dsts[firstpth][2]
                    ps_fft[firstpth]=ps_fft[firstpth]
                      +lk_ffts[firstpth][2]
                end
                else do // path_cnt > 1
                    // loop over paths to sum up denominator
                    dstdenom=0
                    fftdenom=0
                    for jj=1 to did_k do
                        if lk_dsts[jj][2]>0 then do
                        // link is used for path jj
                            dstdenom=dstdenom+min_dist/l_dist[jj]
                            fftdenom=fftdenom+min_fft/l_fft[jj]
                        end
                    end // for jj loop to sum denominator
                    // loop over paths to add arc-specific parts
                    // of path size sum
                    for jj=1 to did_k do
                        if lk_dsts[jj][2]>0 then do

                        // changed so that link size contributions
                        // sum to 1

                            ps_dst[jj]=ps_dst[jj]+lk_dsts[jj][2]*
                                       (min_dist/l_dist[jj])/dstdenom
                            ps_fft[jj]=ps_fft[jj]+lk_ffts[jj][2]*
                                       (min_fft / l_fft[jj])/fftdenom
                        end // if link used for path jj
```

```
                        end // for jj loop
                    end // else path_cnt>1 block
                end // if ... else regular link
            end // if path_cnt>0 block

            link_id=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
            link_cnt=link_cnt+1
            UpdateProgressBar("Calculating Path Sizes",
                               Floor(link_cnt/tot_links*100))
        end // loop over links
        DestroyProgressBar()

        // now finish the path size calculation by normalizing
        // by the constants
        // and put the path sizes in an array to send to the skim table
        for jj=1 to did_k do
            ps_dst[jj]=ps_dst[jj]/(l_dist[jj])
            psdstarr[jj][2]=ps_dst[jj]

            ps_fft[jj]=ps_fft[jj]/(l_fft[jj])
             psfftarr[jj][2]=ps_fft[jj]
        end // calculate and prepare path size loop

        // put path sizes in the skim table

        opdstarr=Subarray(psdstarr,1,did_k)
        opfftarr=Subarray(psfftarr,1,did_k)

        SetRecordValues(skm_view,orgskmid,opdstarr)
        SetRecordValues(skm_view,orgskmid,opfftarr)

        SetRecordValues(skm_view,orgskmid,{{"PS_Def",1}})

    end // path size branch

    if printdirs=1 then do
        timenow=GetDateAndTime()
        WriteLine(prn_file,"Run finished at "+timenow)
        CloseFile(prn_file)
    end

    SetMap("90 UTPS Hwy Asgt")
    SetLayer(rt_layr)

    // get into to write to summary file
    // obsno,best ovl,k found,eliminated link

    if printsumm=1 then
        WriteLine(summ_file,base_name+","+String(max_ovlap[1]*100)+","
                            +String(best_k[1])
                            +","+String(best_rem_lk))


    did_rts=did_rts+1
    if did_rts<tot_rts then
      rt_rh=GetNextRecord(rt_layr+"|"+exo_subname,null,null)
    else rt_rh=null
```

```
end // do over routes


if tot_rts>1 then DestroyProgressBar() // for origins
DisableProgressBar()

// reset these to the Caliper defaults
EnableProgressBar("Status ...",1)
DisableProgressBar()

SetLayer(lk_layr)
SelectNone("tmp_sp")
SelectNone("elim_lk")


// ************************************
// close off the print file
// ************************************

if printsumm=1 then do
    WriteLine(summ_file,"")
    timenow=GetDateAndTime()
    WriteLine(summ_file,"Run finished at "+timenow)
    WriteLine(summ_file,"")
    CloseFile(summ_file)
end


if going_home=1 then
ShowMessage("K Different Paths Macro finished successfully!"+
            "\nIf you need to use Dr. Doolittle, just close TransCAD."+
            "\nThere's no need to save the files; however, you may want"+
            "\nto reboot to clear up memory.")
else
ShowMessage("K Different Paths Macro finished successfully!\n")

endMacro

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Macro "ClearLP" (full)
    lk_view="K40 Links"
    lk_layr="90 UTPS Hwy Asgt"

    max_k=40

    for kj=1 to max_k do
        if kj=1 then do
            lpdstarr={{"P1_DIST",0}}
            lpfftarr={{"P1_FFT" ,0}}
        end
        else do
            ins_arr={{"P"+String(kj)+"_DIST",0}}
            lpdstarr=InsertArrayElements(lpdstarr,kj,ins_arr)
            ins_arr={{"P"+String(kj)+"_FFT" ,0}}
            lpfftarr=InsertArrayElements(lpfftarr,kj,ins_arr)
        end
    end

    SetLayer("90 UTPS Hwy Asgt")

    tot_links=GetRecordCount(lk_view,null)

    link_rh=GetFirstRecord("90 UTPS Hwy Asgt|",null)
    link_cnt=0
    last_pct=0

    CreateProgressBar("Clearing Link Flags","False")
    while (link_rh <> null) and (link_cnt<tot_links) do
        SetRecord(null, link_rh)

        val_temp=GetRecordValues(lk_view,link_rh,{"KS_USED"})
        val_ks_used=val_temp[1][2]

        if (full>0) or (val_ks_used>0) then do
            SetRecordValues(lk_view,link_rh,lpdstarr)
            SetRecordValues(lk_view,link_rh,lpfftarr)
            SetRecordValues(lk_view,link_rh,{{"KS_USED",0}})
        end

        link_rh=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
        link_cnt=link_cnt+1
        if Floor(link_cnt/tot_links*100)>last_pct then do
            UpdateProgressBar("Cleared "+String(link_cnt)+" Link Flags",
                              Floor(link_cnt/tot_links*100))
            last_pct=Floor(link_cnt/tot_links*100)
        end
    end // link_rh loop
    DestroyProgressBar()
EndMacro // ClearLP
```

C.3 Macro to Draw Random Link Impedances

```
/////////////////////////////////////////////////////////
// z:/sramming/files/tc_data/macros/rndlkimp.rsc
//
// Scott Ramming
// W 3 May 2000
// v3.0 M 22 May 2000 - add output of link IDs for later loading
// v4.0 W 16 Aug 2000 - change list file output -- add one field =
//   label imp w/ turn pen (v label skim)
//
// Go through each survey route, build the shortest
// label path from O to D, find how much that overlaps
// with the survey path, and write that to a file,
// preferably for Excel import later
//
// Note: IDs are integers
//       Record Handles appear to be strings
//////////////////////////////////////////////////////////

Macro "DrawRandomLinkImps"

loop_info={{  2039806,1.38809,2*0.426655,60,
             {"IMP1" ,"IMP2" ,"IMP3" ,"IMP4",
              "IMP5" ,"IMP6" ,"IMP7" ,"IMP8"}},
           {  5081696,1.38809,2*0.426655,60,
             {"IMP9" ,"IMP10","IMP11","IMP12",
              "IMP13","IMP14","IMP15","IMP16"}},
           {  7399870,1.38809,2*0.426655,60,
             {"IMP17","IMP18","IMP19","IMP20",
              "IMP21","IMP22","IMP23","IMP24"}},
           {  2537338,1.38809,2*0.426655,60,
             {"IMP25","IMP26","IMP27","IMP28",
              "IMP29","IMP30","IMP31","IMP32"}},
           {021201684,1.38809,2*0.426655,60,
             {"IMP33","IMP34","IMP35","IMP36",
              "IMP37","IMP38","IMP39","IMP40"}},
           {  5121739,1.38809,2*0.426655,60,
             {"IMP41","IMP42","IMP43","IMP44",
              "IMP45","IMP46","IMP47","IMP48"}}}
// format: an array of control arrays
// each control array contains:
// seed,mean,std,draws,array of variable names

tot_loops=ArrayLength(loop_info)

print_dir="d:\\users\\sramming\\tc\\newmystc\\"

lk_layr="90 UTPS Hwy Asgt"
lk_view="New Mystic Link Data"

SetMap(lk_layr)
SetLayer(lk_layr)

EnableProgressBar("Status",3)
```

```
CreateProgressBar("Processing first of "+String(tot_loops)+" control
loops.","False")

// array of IDs of typical links, for printed output
typ_lk_ids={2549, 3634, 3697, 4890, 6846, 6891, 8977, 9265,
            9347, 9681, 10911, 11112, 11698, 16130}

for at_loop=1 to tot_loops do

    loop_parms=loop_info[at_loop]
    rnd_seed=loop_parms[1]

    // set factor_mean & factor_std
    fac_mean=loop_parms[2]
    fac_std =loop_parms[3]
    udrawspern=loop_parms[4]

    var_arr=loop_parms[5]
    tot_lbls=ArrayLength(var_arr)
    for kk=1 to tot_lbls do
        if kk=1 then do
            imp_arr={{var_arr[1],0}}
            fac_arr={1}
        end
        else do
            ins_arr={{var_arr[kk],0}}
            imp_arr=InsertArrayElements(imp_arr,kk,ins_arr)
            fac_arr=InsertArrayElements(fac_arr,kk,{1})
         end
    end // loop over labels

    // open print file
    lst_fn="v2r"+Substring(String(rnd_seed),1,5)+".lst"
    lst_file=OpenFile(print_dir+lst_fn,"w")

    WriteLine(lst_file,"Generate Random Link Impedances")
    WriteLine(lst_file,"Scott Ramming - October 2000")
    WriteLine(lst_file,"")
    timenow=GetDateAndTime()
    WriteLine(lst_file,"Run started at "+timenow)
    WriteLine(lst_file,"Random numbers seeded with "
                       +String(rnd_seed))
    WriteLine(lst_file,"Impedances 1-8 based on C_TIME")
    WriteLine(lst_file,"Average of impedance factor = "
                       +String(fac_mean))
    WriteLine(lst_file,"Std. dev. of impedance factor = "
                       +String(fac_std))
    WriteLine(lst_file,"Using "+String(udrawspern)
                       +" Uniform draws per Normal draw")
    WriteLine(lst_file,"")
    hdr_line="Link_ID,Name,C_Time"
    for kk=1 to tot_lbls do
        hdr_line=hdr_line+","+var_arr[kk]
    end
    WriteLine(lst_file,hdr_line)

    SetRandomSeed(rnd_seed)
```

```
tot_recs=GetRecordCount(lk_view,null)
did_recs=0
last_pct=0

CreateProgressBar("Drew Impedance for 0 of "
                  +String(tot_recs)+" Links","False")

rec_rh = GetFirstRecord(lk_view+"|", null)
while (rec_rh <> null) and (did_recs<=tot_recs) do
    // select record
    SetRecord(null, rec_rh)
    ct=lk_view.C_TIME
    lk_id=lk_view.ID

    if lk_view.CENT_CONN>0 or lk_view.CAPACITY=999999 then do
    // copy c_time to imp1-8
        for ii=1 to tot_lbls do
            imp_arr[ii][2]=ct
        end
    end
    else do
        for ii=1 to tot_lbls do // make draws for each
            fac_arr[ii]=fac_mean
              +fac_std*RunMacro("DrawStdNormal",udrawspern)
            imp_arr[ii][2]=Max(ct*fac_arr[ii],0.01)
            // impedance should always be at least slightly positive
        end
        // if link ID in typical array
        if ArrayPosition(typ_lk_ids,{lk_id},null)>0 then do
            prt_string=String(lk_id)+","+lk_view.NAME+","+String(ct)
            for ii=1 to tot_lbls do
                prt_string=prt_string+","+String(imp_arr[ii][2])
            end
            WriteLine(lst_file,prt_string)
        end
    end // else real link
    SetRecordValues(lk_view,rec_rh,imp_arr)

    did_recs=did_recs+1
    if Floor(did_recs/tot_recs*100)>last_pct then do
        last_pct=Floor(did_recs/tot_recs*100)
        UpdateProgressBar("Drew Impedance for "+String(did_recs)+" of "
            +String(tot_recs)+" Links",last_pct)
    end
    last_rh = rec_rh

    rec_rh = GetNextRecord(lk_view+"|", last_rh, null)
end // loop over records

DestroyProgressBar()

WriteLine(lst_file,"")
timenow=GetDateAndTime()
WriteLine(lst_file,"Run finished at "+timenow)
WriteLine(lst_file,"")
CloseFile(lst_file)
```

```
    UpdateProgressBar("Completed "+String(at_loop)+" of "
                      +String(tot_loops)+" control loops.",
                      Floor(100*at_loop/tot_loops))

end // big control loop

DestroyProgressBar()
DisableProgressBar()

ShowMessage("Draw Random Link Impedance Macro Finished.")

endMacro

// **************************************************************************

Macro "DrawStdNormal"(ptsperdraw)
sum=0
for ij=1 to ptsperdraw do
    sum=sum + RandomNumber()
end
// take average
sum=sum/ptsperdraw
// now mean=0.5
// var=1/12n
// normalize
sum=(sum-0.5)*Sqrt(12*ptsperdraw)
Return(sum)
endMacro

// **********************************************************************
// The following two macros are included for debugging and
// expository purposes
// **********************************************************************

Macro "TestNDrawPts"
ShowMessage("Starting TestNDrawPts")
// set array of pts to test
// ptsarray={1, 6, 12, 30, 60}
ptsarray={30}
cdfpts=20000
rnd_seed=2039806 // or any other value
print_dir="d:\\users\\sramming\\tc\\tmp3\\"

lst_file=OpenFile(print_dir+"d30_20k.lst","w")
timenow=GetDateAndTime()
WriteLine(lst_file,"Run started at "+timenow)
WriteLine(lst_file,"Requested 20,000 Normal Draws using 30 Uniforms Per
Normal")
WriteLine(lst_file,"")

for tr=1 to ArrayLength(ptsarray) do
    // open print file(s)
    pfilename=print_dir+"timen_"+String(ptsarray[tr])
               +".lst"
    prn_file=OpenFile(pfilename,"w")
    WriteLine(prn_file,"Test Normal (Gaussian) Draws")
    WriteLine(prn_file,"Scott Ramming - September 2000")
    SetRandomSeed(rnd_seed)
    WriteLine(prn_file,"Random numbers seeded with "
       +String(rnd_seed))
    WriteLine(prn_file,"")

    for dd=1 to cdfpts do
        val=RunMacro("DrawStdNormal",ptsarray[tr])
        WriteLine(prn_file,String(val))
    end // for dd
    WriteLine(prn_file,"")
    // close files
    CloseFile(prn_file)
end // for tr

timenow=GetDateAndTime()
WriteLine(lst_file,"Run finished at "+timenow)
WriteLine(lst_file,"")
CloseFile(lst_file)

ShowMessage("Ending TestNDrawPts")
endMacro
// look at results in Excel charts

// **************************************************************************

Macro "TestOneDraw"
    val=RunMacro("DrawStdNormal",12)
    ShowMessage("The value drawn was "+String(val)+".")
endMacro
```

C.4 Macro to Minimize Multiple Labels and Output LPI Matrix

```
// --------------------------------------------------------------------------
// this is file://magellancee.mit.edu/d|users/sramming/tcmac/mlbl_csg.rsc
//
// Scott Ramming
// January 2001
//
// This macro allows you to specify an array of labels (variable names)
// and a selection set of survey routes. For each route, TransCAD will
// determine the O and D, and generate alternative paths according to
// minimization of each label. TC will check for duplicates and at the
// end, indicate which generated path most closely resembles (distance
// overlap) the survey path.
// --------------------------------------------------------------------------

Macro "MultiLabelCSG"

// --------------------------------------------------------------------------
// Run-related variables:

label_array={"DISTANCE","C_TIME","FF_TIME","IMP1","IMP2","IMP3","IMP4",
             "IMP5","IMP6","IMP7","IMP8","IMP9","IMP10","IMP11","IMP12",
             "IMP13","IMP14","IMP15","IMP16","IMP17","IMP18","IMP19",
             "IMP20","IMP21","IMP22","IMP23","IMP24","IMP25","IMP26",
             "IMP27","IMP28","IMP29","IMP30","IMP31","IMP32","IMP33",
             "IMP34","IMP35","IMP36","IMP37","IMP38","IMP39","IMP40",
             "IMP41","IMP42","IMP43","IMP44","IMP45","IMP46","IMP47",
             "IMP48"}
max_k=52 // the most spaces in the skim and link-path files

debug=0 // 1

rt_set="Selection"
// name of selection set for which to create paths
// from O to D by minimizing labels

printout=1  // create a summary file of rt ids, # paths generated, best match
save_lkid=1 // create a summary file of link ids for all paths generated
save_dirs=1 // create text files (one per route) for directions of each path

calc_ps=0
// 1 to calculate path sizes
// 0 to skip
ps_def=2
ps_param=99

// --------------------------------------------------------------------------
// Project-related variables:

temp_dir ="d:\\users\\sramming\\tc\\tmp3\\"
print_dir="d:\\users\\sramming\\tc\\newmystc\\paths\\"
link_dir ="d:\\users\\sramming\\tc\\newmystc\\paths\\"

rt_view="North Shore Survey R"
rt_layr="North Shore Survey R"

lk_view="New Mystic K52 Data"
lk_layr="90 UTPS Hwy Asgt"

nd_layr="Endpoints"
nd_view="New Mystic Node Data"

skm_view="New Mystic Skim Table"

net_file="d:\\users\\sramming\\tc\\newmystc\\nmystic1.net"
pen_file="d:\\users\\sramming\\tc\\ftp3f5.dbf"

// --------------------------------------------------------------------------
// auxiliary variables and TransCAD initialization:

ntfh=ReadNetwork(net_file)

shared d_NetInfo

dim sel_lnks[200]
dim label_pos[52]
dim rt_ovls[52]
dim ps_args[8]

RunMacro("TCU set network",net_file,ntfh,null)
```

```
net_hand=d_NetInfo[2][2]

d_NetInfo[3][1]=1
d_NetInfo[3][2]=0
d_NetInfo[3][3]=1

net_lk_vars=NetworkLinkVarNames(net_hand)

orig_cc_pos=ArrayPosition(net_lk_vars,{"orig_cc"},null)
dest_cc_pos=ArrayPosition(net_lk_vars,{"dest_cc"},null)
real_rd_pos=ArrayPosition(net_lk_vars,{"real_rd"},null)
cc_pos=ArrayPosition(net_lk_vars,{"Cent_Conn"},null)

lk_flds=GetFields(lk_view,"Numeric")

d_NetInfo[4]={cc_pos,1,null}

pen_file_parts=ParseString(pen_file,"\\")
short_pen_file=pen_file_parts[ArrayLength(pen_file_parts)]

d_NetInfo[6]={null,{pen_file,short_pen_file,"DBASE"},
              0,0,0,0}

skim_arr={ {"distance","sum"}, {"ff_time","sum"}, {"c_time","sum"},
           {"one","sum"}, {"freeway","sum"},{"fwy_time","sum"},
           {"expwy","sum"}, {"expwy_time","sum"}, {"unsafe","sum"},
           {"unsf_time","sum"}, {"num_route","sum"}, {"numrt_time","sum"},
           {"toll","sum"} }
sum_arr={ "SUMDISTANC", "SUMFF_TIME", "SUMC_TIME",
          "SUMONE", "SUMFREEWAY", "SUMFWY_TIM",
          "SUMEXPWY", "SUMEXPWY_T", "SUMUNSAFE",
          "SUMUNSF_TI", "SUMNUM_ROU", "SUMNUMRT_T", "SUMTOLL" }

tot_lbls=ArrayLength(label_array)

// check that all labels appear in the network
curr_lbl=1
removed_lbls=0

while (curr_lbl<=tot_lbls) do

label_pos[curr_lbl]=ArrayPosition(net_lk_vars,{label_array[curr_lbl]},null)
    if label_pos[curr_lbl]=0 then do // remove from label_array
        label_array=ExcludeArrayElements(label_array,curr_lbl,1)
        tot_lbls=tot_lbls-1
        removed_lbls=removed_lbls+1
    end
    else do // it is found
        curr_lbl=curr_lbl+1
    end
end

SetLayer(rt_layr)
tot_rts=GetSetCount(rt_set)
did_rts=0

// **************************************************************************
// open the summary file and print out run parameters
// ***************************************************************************

// summary file has extension .lst
if printout=1 then do
    summ_fn=print_dir+Substring(rt_set,1,8)+".lst"
    prn_file=OpenFile(summ_fn,"w")
    WriteLine(prn_file,"Multi-Label Choice Set Generation - Version 1.0")
    WriteLine(prn_file,"Scott Ramming - January 2001")
    WriteLine(prn_file,"")
    timenow=GetDateAndTime()
    WriteLine(prn_file,"Run started at "+timenow)
    WriteLine(prn_file,"")
    WriteLine(prn_file,"Run options:")
    WriteLine(prn_file,String(tot_lbls+removed_lbls)
               +" label paths requested")
    if removed_lbls>0 then do
        WriteLine(prn_file,String(removed_lbls)
                    +" not found in network (removed)")
        WriteLine(prn_file,String(tot_lbls)+" accepted for processing")
    end
    WriteLine(prn_file,"")
    if calc_ps>0 then do
        WriteLine(prn_file,"Path Size calculations requested")
        WriteLine(prn_file,"Using definition "+String(ps_def)
                  +" and parameter "+String(ps_param))
        WriteLine(prn_file,"")
    end
    WriteLine(prn_file,"RouteName,RoutesBuilt,BestMatch,BestOverlap")
end

// ***************************************************************************
// open the link file and print out run parameters
// ***************************************************************************

// link file has extension .csv

// proposed format:
//
// MultiLabelCSG 1.0                         // version ident
// rt_set,tot_rts,tot_lbls,max_k             // run params
// label_array                               // list of labels
// route_name,orig_rh,dest_rh                // route header
// found_k,curr_lbl,label_array,num_links    // unique path header
// link ID, direction flag                   // link info
// [repeats]
// -1                                        // verify end of link info
// [repeat from unique path header]
// -9                                        // end of unique routes
//                                        (total number unknown beforehand)
// min(found_k,max_k),k of best match,best match overlap
                                             // path gen summary for route
// -19                                       // verify end of route info
// -39                                       // verify end of routes and
                                             // therefore of file

if save_lkid=1 then do
```

```
    link_fn=print_dir+Substring(rt_set,1,8)+".csv"
    lk_fh=OpenFile(link_fn,"w")
    WriteLine(lk_fh,"MultiLabelCSG 1.0")
    WriteLine(lk_fh,rt_set+","+String(tot_rts)+","+String(tot_lbls)+","
                    +String(max_k))
    for lbl=1 to tot_lbls do
        WriteLine(lk_fh,label_array[lbl])
    end
end

// **************************************************************************
// begin the big loop for each survey route
// **************************************************************************

EnableProgressBar("Status ...",3)
CreateProgressBar("Generated Choice Set for 0 of "+String(tot_rts)
                  +" Survey Repondents.","False")

rt_rh = GetFirstRecord(rt_set, null)
// *************************************************
// loop over selected survey routes
// *************************************************
while (rt_rh <> null) & (did_rts<tot_rts)  do
    // select record
    SetRecord(null, rt_rh)
    SelectNone("mac_temp")
    SelectRecord("mac_temp")

    rt_name=rt_view.Route_Name
    base_name=Substring(rt_name,3,4)

    if base_name="" then do
        rt_name=Format(rt_view.Route_ID,"0000")
    end

    obsno=S2I(base_name)

    // code to find origin and destination
    lnk_tmp=GetRouteLinks(rt_view,rt_name)
    nrlinks=ArrayLength(lnk_tmp)
    occ_id=lnk_tmp[1][1]
    dcc_id=lnk_tmp[nrlinks][1]

    SetLayer(lk_layr)
    occ_end=GetEndpoints(occ_id)
    dcc_end=GetEndpoints(dcc_id)

    SetLayer(nd_layr)
    occa_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{occ_end[1]},
                         {"Exact","True"})
    SetRecord(nd_layr,occa_rh)
    if nd_view.CENTROID>0 then do
        orig_id=occ_end[1]
        SelectNone("curr_orig")
        SelectRecord("curr_orig") // for debugging only
    end
    else do
        orig_id=occ_end[2]
        occb_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{occ_end[2]},
                             {"Exact","True"})
        SetRecord(null,occb_rh)
        SelectNone("curr_orig")
        SelectRecord("curr_orig")
    end

    dccb_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{dcc_end[2]},
                         {"Exact","True"})
    SetRecord(nd_layr,dccb_rh) // since most dest are new
    if nd_view.CENTROID>0 then do
        dest_id=dcc_end[2]
        SelectNone("curr_dest")
        SelectRecord("curr_dest")
    end
    else do
        dest_id=dcc_end[1]
        dcca_rh=LocateRecord(nd_layr+"|",nd_layr+".ID",{dcc_end[1]},
                             {"Exact","True"})
        SetRecord(null,dcca_rh)
        SelectNone("curr_dest")
        SelectRecord("curr_dest")
    end

    // write OD/respondent header to link ID file
    if save_lkid=1 then do
        WriteLine(lk_fh,rt_name+","+String(orig_id)+","+String(dest_id))
    end

    if save_dirs=1 then do
        // open the directions file
        dir_file=OpenFile(print_dir+"dir"+base_name+".txt","w")
        WriteLine(dir_file,"Multi-Label Path Generation - Version 1.0")
        WriteLine(dir_file,"Scott Ramming - January 2001")
        WriteLine(dir_file,"")
        WriteLine(dir_file,"See file "+Substring(rt_set,1,8)
                          +".lst for run options.")
        WriteLine(dir_file,"Directions for survey respondent "+rt_name+".")
        WriteLine(dir_file,"")
        timenow=GetDateAndTime()
        WriteLine(dir_file,"Path generation started at "+timenow)
        WriteLine(dir_file,"")
    end
    // ********************************************
    // prepare for loop over labels
    // ********************************************

    // clear the link-path matrix
    // ==========================
    SetLayer(lk_layr)
    full=0
    RunMacro("ClearLP",full)

    curr_lbl=1
    found_k=0
    CreateProgressBar("Finding the first path.","False")
```

```
// ********************************************
// loop over the labels
// ********************************************
while (curr_lbl<=tot_lbls)&(found_k<=max_k) do

    // get the shortest path
    // =====================

    curr_sp=ShortestTurnPath(net_hand,orig_id,dest_id,label_pos[curr_lbl],
            {{"Update Penalties",{d_NetInfo[6][3],d_NetInfo[6][4],
                                  d_NetInfo[6][5],d_NetInfo[6][6],
                                  null,d_NetInfo[6][2][1]}},
             {"Link Type",d_NetInfo[4][1]},
             {"Connector",d_NetInfo[4][2]}})

    lbl_imp=curr_sp[1]

    nrlinks=ArrayLength(curr_sp[2]) // needed for lk_id file
    sel_lnk2=curr_sp[2]
    sp_dirs=curr_sp[3]

    // SelectByIDs("tmp_sp","Several",sel_lnk2)
    // ========================================
    // loop to put SP routes into two selection sets baed on direction
    // ===============================================================
    SelectNone("tmp_sp")
    SelectNone("tmp_sp1")
    for aa=1 to nrlinks do
        if sp_dirs[aa]=0 then do
            SelectByIDs("tmp_sp","More",{sel_lnk2[aa]})
        end
        else do // sp_dirs[aa]=1
            SelectByIDs("tmp_sp1","More",{sel_lnk2[aa]})
        end
    end

    uniq=1
    if curr_lbl>1 then do
        // test for uniqueness
        if found_k=1 then do
            comp_arr={ {"distance","sum"}, {"p1_dist","sum"} }
            comp_flds={"SUMDISTANC","SUMP1_DIST"}
            cmp1_arr={ {"distance","sum"}, {"p1_dist1","sum"} }
            cmp1_flds={"SUMDISTANC","SUMP1_DIST"}
            // suffix 1 gets truncated because there's no
            // p1_dist in the same view to sum
        end
        else do // add another
            ins_arr={ {"p"+String(found_k)+"_dist","sum"} }
            comp_arr=InsertArrayElements(comp_arr,found_k+1,ins_arr)
            ins_arr={ {"p"+String(found_k)+"_dist1","sum"} }
            cmp1_arr=InsertArrayElements(cmp1_arr,found_k+1,ins_arr)
            kw=1
            if found_k>9 then kw=2
            ins_arr={"SUMP"+String(found_k)+Substring("_DIST",1,6-kw)}
            comp_flds=InsertArrayElements(comp_flds,found_k*2,ins_arr)
            ins_arr={"SUMP"+String(found_k)+Substring("_DIST",1,6-kw)}
            cmp1_flds=InsertArrayElements(cmp1_flds,found_k*2,ins_arr)
        end

        tmp_suffix=Substring("0123456789abcdefghijklmnopqrstuvwxyz",
            Floor(curr_lbl/32)+1,1)+
            Substring("0123456789abcdefghijklmnopqrstuvwxyz",
            RealToInt(Mod(curr_lbl,32))+1,1)
        tmp_view="r"+base_name+"i"+tmp_suffix
        tmp_fnam=temp_dir+tmp_view+".dbf"
        tm1_view="r"+base_name+"j"+tmp_suffix
        tm1_fnam=temp_dir+tm1_view+".dbf"

        AggregateTable(tmp_view,lk_view+"|tmp_sp","dBASE",
              tmp_fnam,"ONE",comp_arr, null )

        tmp_rh=GetFirstRecord(tmp_view+"|",null)
        // first and only record
        comp_dsts=GetRecordValues(tmp_view,tmp_rh,comp_flds)

        CloseView(tmp_view)

        if debug=1 then do
            // RunDBox("G30 Show Array","comp_arr" ,comp_arr )
            // RunDBox("G30 Show Array","comp_flds",comp_flds)
            // RunDBox("G30 Show Array","comp_dsts",comp_dsts)
        end

        // now distance of current generated path is in comp_dsts[1][2]
        // distance of earlier paths is comp_dsts[k+1][2]
        // for forward links only

        nblinks=GetSetCount("tmp_sp1")
        if nblinks=0 then do // added later - no indent

        kk=1
        while (kk<=found_k) and (uniq=1) do
            if Abs(comp_dsts[1][2]-comp_dsts[kk+1][2])<0.01
            then do
                uniq=0
                if debug=1 then do
                    diff=comp_dsts[1][2]-comp_dsts[kk+1][2]
                    ShowMessage("Subject distance comp_dsts[1][2] = "
                        +String(comp_dsts[1][2])
                        +"\nComparison distance comp_dsts["
                        +String(kk+1)+"][2] = "
                        +String(comp_dsts[kk+1][2])
                        +"\ndifference = "+String(diff)
                        +"\nkk = "+String(kk)+"\nuniq = "
                        +String(uniq) )
                end
            end
            else do
                if debug=1 then do
                    diff=comp_dsts[1][2]-comp_dsts[kk+1][2]
                    ShowMessage("Subject distance comp_dsts[1][2] = "
                        +String(comp_dsts[1][2])
                        +"\nComparison distance comp_dsts["
```

```
                        +String(kk+1)+"][2] = "
                        +String(comp_dsts[kk+1][2])
                        +"\ndifference = "+String(diff)
                        +"\nkk = "+String(kk)+"\nuniq = "
                        +String(uniq) )
                end
                kk=kk+1
            end // else distances are different
        end // while testing for uniqueness

        end // if nblinks=0 - no indent in block
        else do // nblinks>=1 - more common case - no indent in block

        AggregateTable(tm1_view,lk_view+"|tmp_sp1","dBASE",
              tm1_fnam,"ONE",cmp1_arr, null )

        tm1_rh=GetFirstRecord(tm1_view+"|",null)
        // first and only record
        cmp1_dsts=GetRecordValues(tm1_view,tm1_rh,cmp1_flds)

        CloseView(tm1_view)

        if debug=1 then do
            // RunDBox("G30 Show Array","cmp1_arr" ,cmp1_arr )
            // RunDBox("G30 Show Array","cmp1_flds",cmp1_flds)
            // RunDBox("G30 Show Array","cmp1_dsts",cmp1_dsts)
        end

        kk=1
        while (kk<=found_k) and (uniq=1) do
            if Abs(comp_dsts[1][2]-comp_dsts[kk+1][2])<0.01 and
               Abs(cmp1_dsts[1][2]-cmp1_dsts[kk+1][2])<0.01
            then do
                uniq=0
            end
            else do
                kk=kk+1
            end // else distances are different
        end // while testing for uniqueness

        end // if nblinks>=1 - no indent in block
    end // if curr_lbl>1

    if uniq=1 then do
        if found_k=max_k then do
            // error handling if already found max_k
            // basically need to skip to incrementing found_k
            curr_lbl=tot_lbls
            // shouldn't happen, so I'm not coding much here
        end
        else do
            found_k=found_k+1

            // update all used links set for path size calculation
            // ================================================
            if calc_ps>0 then do
                if curr_lbl=1 then SelectNone("tmp_used")
                SetOR("tmp_used",{"tmp_used","tmp_sp","tmp_sp1"})
            end

            // set link flags
            // ==============
            for curr_link=1 to nrlinks do
                link_id=sel_lnk2[curr_link]
                currlk_rh=LocateRecord(lk_view+"|","ID",
                     {link_id},{"Exact","True"})
                SetRecord(null,currlk_rh)
                if sp_dirs[curr_link]=0 then do
                    SetRecordValues(lk_view,currlk_rh,
                      { {"P"+String(found_k)+"_DIST",lk_view.distance},
                        {"P"+String(found_k)+"_FFT", lk_view.ff_time} })
                end
                else do
                    SetRecordValues(lk_view,currlk_rh,
                      { {"P"+String(found_k)+"_DIST1",lk_view.distance},
                      {"P"+String(found_k)+"_FFT1", lk_view.ff_time} })
                end
                lk_view.ks_used=lk_view.ks_used+1
            end // link_id loop to increase impedances

            // copy skims
            // ==========
            tmp_suffix=Substring("0123456789abcdefghijklmnopqrstuvwxyz",
                Floor(found_k/32)+1,1)+
                Substring("0123456789abcdefghijklmnopqrstuvwxyz",
                RealToInt(Mod(found_k,32))+1,1)
            tmp_view="s"+base_name+"k"+tmp_suffix
            tmp_fnam=temp_dir+tmp_view+".dbf"
            tm1_view="s"+base_name+"m"+tmp_suffix
            tm1_fnam=temp_dir+tm1_view+".dbf"

            AggregateTable(tmp_view,lk_view+"|tmp_sp","dBASE",
                  tmp_fnam,"ONE",skim_arr, null )

            tmp_rh=GetFirstRecord(tmp_view+"|",null)
            // first and only record
            skim_tmp=GetRecordValues(tmp_view,tmp_rh,sum_arr)

            CloseView(tmp_view)

            AggregateTable(tm1_view,lk_view+"|tmp_sp1","dBASE",
                  tm1_fnam,"ONE",skim_arr, null )
            record_ck=GetRecordCount(tm1_view,null)

            if record_ck=1 then do // add skims together
                tm1_rh=GetFirstRecord(tm1_view+"|",null)
                skm1_tmp=GetRecordValues(tm1_view,tm1_rh,sum_arr)

                skim_put=
                {{"p"+String(found_k)+"dist",
                  skim_tmp[1][2] +skm1_tmp[1][2]},
                 {"p"+String(found_k)+"fft" ,
                  skim_tmp[2][2] +skm1_tmp[2][2]},
                 {"p"+String(found_k)+"ct"  ,
                      skim_tmp[3][2] +skm1_tmp[3][2]},
                     {"p"+String(found_k)+"lnks",
                      skim_tmp[4][2] +skm1_tmp[4][2]},
                     {"p"+String(found_k)+"fwyl",
                      skim_tmp[5][2] +skm1_tmp[5][2]},
                     {"p"+String(found_k)+"fwyt",
                      skim_tmp[6][2] +skm1_tmp[6][2]},
                     {"p"+String(found_k)+"expl",
                      skim_tmp[7][2] +skm1_tmp[7][2]},
                     {"p"+String(found_k)+"expt",
                      skim_tmp[8][2] +skm1_tmp[8][2]},
                     {"p"+String(found_k)+"unsl",
                      skim_tmp[9][2] +skm1_tmp[9][2]},
                     {"p"+String(found_k)+"unst",
                      skim_tmp[10][2]+skm1_tmp[10][2]},
                     {"p"+String(found_k)+"nmrl",
                      skim_tmp[11][2]+skm1_tmp[11][2]},
                     {"p"+String(found_k)+"nmrt",
                      skim_tmp[12][2]+skm1_tmp[12][2]},
                     {"p"+String(found_k)+"toll",
                      skim_tmp[13][2]+skm1_tmp[13][2]}}
                end // forward and back link skims added
                else do // no backwards links
                    skim_put={{"p"+String(found_k)+"dist",skim_tmp[1][2]},
                              {"p"+String(found_k)+"fft" ,skim_tmp[2][2]},
                              {"p"+String(found_k)+"ct"  ,skim_tmp[3][2]},
                              {"p"+String(found_k)+"lnks",skim_tmp[4][2]},
                              {"p"+String(found_k)+"fwyl",skim_tmp[5][2]},
                              {"p"+String(found_k)+"fwyt",skim_tmp[6][2]},
                              {"p"+String(found_k)+"expl",skim_tmp[7][2]},
                              {"p"+String(found_k)+"expt",skim_tmp[8][2]},
                              {"p"+String(found_k)+"unsl",skim_tmp[9][2]},
                              {"p"+String(found_k)+"unst",skim_tmp[10][2]},
                              {"p"+String(found_k)+"nmrl",skim_tmp[11][2]},
                              {"p"+String(found_k)+"nmrt",skim_tmp[12][2]},
                              {"p"+String(found_k)+"toll",skim_tmp[13][2]}}
                end // no backwards links
                CloseView(tm1_view)

                SetView(skm_view)
                orgskmrh=LocateRecord(skm_view+"|","OBSNO",
                         {StringToInt(base_name)},{"Exact","True"})
                SetRecordValues(skm_view,orgskmrh,skim_put)

                SetLayer(lk_layr)
                SetView(lk_view)

                // write link IDs to file
                // =====================
                if save_lkid=1 then do

// format is:
// found_k,curr_lbl,label_array,num_links    // unique path header
// link ID, direction flag                   // link info
// [repeats]
// -1                                        // verify end of link info
```

```
                WriteLine(lk_fh,String(found_k)+","+String(curr_lbl)+","
                          +label_array[curr_lbl]+","+String(nrlinks))
                for curr_link=1 to nrlinks do
                    WriteLine(lk_fh,String(sel_lnk2[curr_link])+","
                              +String(sp_dirs[curr_link]) )
                end
                WriteLine(lk_fh,"-1")
            end

            // copy directions
            // ===============
            if save_dirs=1 then do
                tmp_suffix=
                    Substring("0123456789abcdefghijklmnopqrstuvwxyz",
                    Floor(found_k/32)+1,1)+
                    Substring("0123456789abcdefghijklmnopqrstuvwxyz",
                    RealToInt(Mod(found_k,32))+1,1)
                // unchanged from copying skims above
                tmp_view="d"+base_name+"k"+tmp_suffix
                tmp_dnam=temp_dir+tmp_view+".txt"

                PathDirections(sel_lnk2,sp_dirs,lk_view+".Name",
                    tmp_dnam,lk_view+".ID")

                // now copy the whole thing over to the big file

                WriteLine(dir_file,"Unique path "+String(found_k)
                          +" , generated by label "+label_array[curr_lbl]
                          +" :")

                WriteLine(dir_file,"(total impedance = "+
                    String(lbl_imp)+".)")

                td_file=OpenFile(tmp_dnam,"r")
                while !FileAtEOF(td_file) do
                    tmp_line=ReadLine(td_file)
                    if StringLength(tmp_line)>0 then
                        WriteLine(dir_file,tmp_line)
                end
                CloseFile(td_file)
                WriteLine(dir_file," ")
            end // save directions

        end // else room for more paths
    end // if unique path

    if found_k<=max_k then do
        UpdateProgressBar("Found "+String(found_k)+" unique paths ("
            +String(curr_lbl)+" of "+String(tot_lbls)+" labels).",
            Floor(100*curr_lbl/tot_lbls))
    curr_lbl=curr_lbl+1
    end

end // loop over labels
DestroyProgressBar()

// close the directions file
```

```
    if save_dirs=1 then do
        timenow=GetDateAndTime()
        WriteLine(dir_file,"Path generation finished at "+timenow)
        WriteLine(dir_file,"")
        CloseFile(dir_file)
    end

    // find the best match
    // select links from survey route
    // =============================
    lnk_tmp=GetRouteLinks(rt_view,rt_name)

    nrlinks=ArrayLength(lnk_tmp)

    for ij=1 to nrlinks do
        sel_lnks[ij]=lnk_tmp[ij][1]
    end

    sel_lnk2=Subarray(sel_lnks,1,nrlinks)

    SelectByIDs("tmp_srv","Several",sel_lnk2)

    // skim PxDIST flags
    // =================
    for curr_lbl=1 to found_k do
        if curr_lbl=1 then do
            comp_arr={ {"distance","sum"}, {"p1_dist","sum"},
                       {"p1_dist1","sum"} }
            comp_flds={"SUMDISTANC","SUMP1_DIST","SUMP1_DIS1"}
        end
        else do // add another
            ins_arr={ {"p"+String(curr_lbl)+"_dist","sum"},
                      {"p"+String(curr_lbl)+"_dist1","sum"} }
            comp_arr=InsertArrayElements(comp_arr,curr_lbl*2+1,ins_arr)
            kw=1
            if curr_lbl>9 then kw=2
            ins_arr={"SUMP"+String(curr_lbl)+Substring("_DIST",1,6-kw),
                     "SUMP"+String(curr_lbl)+Substring("_DIS",1,5-kw)+"1"}
            comp_flds=InsertArrayElements(comp_flds,curr_lbl*2+1,ins_arr)
        end
    end // loop to create skim flags

    tmp_view="r"+base_name+"cmp"
    tmp_fnam=temp_dir+tmp_view+".dbf"

    AggregateTable(tmp_view,lk_view+"|tmp_srv","dBASE",
          tmp_fnam,"ONE",comp_arr, null )

    tmp_rh=GetFirstRecord(tmp_view+"|",null)
    // first and only record
    comp_dsts=GetRecordValues(tmp_view,tmp_rh,comp_flds)

    CloseView(tmp_view)

    // find the max
    // ============
    for curr_lbl=1 to found_k do
        rt_ovls[curr_lbl]=comp_dsts[2*curr_lbl][2]+comp_dsts[2*curr_lbl+1][2]
    end

    rt_ovl2=Subarray(rt_ovls,1,found_k)

    max_ovl=ArrayMax(rt_ovl2)
    best_k=ArrayPosition(rt_ovl2,{max_ovl},null)
    max_ovl_pct=100*max_ovl/comp_dsts[1][2]

    // write to skim file
    // ==================
    SetView(skm_view)
    orgskmrh=LocateRecord(skm_view+"|","OBSNO",
             {StringToInt(base_name)},{"Exact","True"})
    SetRecord(null,orgskmrh)
    skm_view.MATCHES=best_k
    skm_view.KS_BUILT=found_k

    // write to link ID file
    // =====================
    // format:
    // -9                        // end of unique routes
    // min(found_k,max_k),k of best match,best match overlap
                                 // path gen summary for route=OD pair
    // -19                       // verify end of route info

    if save_lkid=1 then do
        WriteLine(lk_fh,"-9")
        WriteLine(lk_fh,String(found_k)+","+String(best_k)+","
                 +String(max_ovl_pct))
        WriteLine(lk_fh,"-19")
    end

    // write info to summary file
    // ==========================
    if printout=1 then do
        WriteLine(prn_file,rt_name+","+String(found_k)+","+String(best_k)
                  +","+String(max_ovl_pct))
    end

    // ====================
    // calculate path sizes
    // ====================
    if calc_ps>0 then do
        // assemble macro arguments into an array
        // arguments are:
        // 1. ps_definition
        //      2=current Li/Lj formulation with exponent parameter
        // 2. ps_parameter
        // 3. name of link view
        // 4. set name with links used (to reduce run time)
        // 5. number of paths produced
        // 6. name of skim view (where path sizes will be stored)
        // 7. observation number
        // 8. flag indicating to calculate distance or fft-based sizes,
        //       or both
```

```
        ps_args={ps_def,ps_param,lk_view,"tmp_used",found_k,skm_view,
                 StringToInt(base_name),3}

        RunMacro("CalcPS",ps_args)
    end

    did_rts=did_rts+1
    UpdateProgressBar("Generated Choice Set for "+String(did_rts)+" of "
                      +String(tot_rts)+" Survey Repondents.",
                      Floor(100*did_rts/tot_rts))

    SetLayer(rt_layr)

    rt_rec = rt_rh

    rt_rh = GetNextRecord(rt_set, rt_rec, null)
end // loop over routes

// close the link ID file
// ======================
if save_lkid=1 then do
    WriteLine(lk_fh,"-39")
    CloseFile(lk_fh)
end

// write run summary and close the summary file
// ============================================
if printout=1 then do
    WriteLine(prn_file,"")
    WriteLine(prn_file,"Built label paths for "+String(did_rts)
              +" survey respondents")
    WriteLine(prn_file,"")
    timenow=GetDateAndTime()
    WriteLine(prn_file,"Run finished at "+timenow)
    WriteLine(prn_file,"")

    CloseFile(prn_file)
end

DestroyProgressBar()
DisableProgressBar()

ShowMessage("Multi-Label Choice Set Generation Macro Completed.")

endMacro

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Macro "ClearLP" (full)
    lk_view="New Mystic K52 Data"
    lk_layr="90 UTPS Hwy Asgt"

    max_k=52

    for kj=1 to max_k do
        if kj=1 then do
            lpdstarr={{"P1_DIST",0},{"P1_DIST1",0}}
            lpfftarr={{"P1_FFT" ,0},{"P1_FFT1" ,0}}
        end
        else do
            ins_arr={{"P"+String(kj)+"_DIST",0},{"P"+String(kj)+"_DIST1",0}}
            lpdstarr=InsertArrayElements(lpdstarr,kj*2-1,ins_arr)
            ins_arr={{"P"+String(kj)+"_FFT" ,0},{"P"+String(kj)+"_FFT1" ,0}}
            lpfftarr=InsertArrayElements(lpfftarr,kj*2-1,ins_arr)
        end
    end

    SetLayer("90 UTPS Hwy Asgt")

    tot_links=GetRecordCount(lk_view,null)

    link_rh=GetFirstRecord("90 UTPS Hwy Asgt|",null)
    link_cnt=0
    last_pct=0

    CreateProgressBar("Clearing Link Flags","False")
    while (link_rh <> null) and (link_cnt<tot_links) do
        SetRecord(null, link_rh)

        val_temp=GetRecordValues(lk_view,link_rh,{"KS_USED"})
        val_ks_used=val_temp[1][2]

        if (full>0) or (val_ks_used>0) then do
            SetRecordValues(lk_view,link_rh,lpdstarr)
            SetRecordValues(lk_view,link_rh,lpfftarr)
            SetRecordValues(lk_view,link_rh,{{"KS_USED",0}})
        end

        link_rh=GetNextRecord("90 UTPS Hwy Asgt|",null,null)
        link_cnt=link_cnt+1
        if Floor(link_cnt/tot_links*100)>last_pct then do
            UpdateProgressBar("Cleared "+String(link_cnt)+" Link Flags",
                              Floor(link_cnt/tot_links*100))
            last_pct=Floor(link_cnt/tot_links*100)
        end
    end // link_rh loop
    DestroyProgressBar()
EndMacro // ClearLP

// %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Macro "CalcPS" (args)
// ----------------------------------------------------------------
// Scott Ramming
// January 2001
//
// This macro calculates path sizes from a predefined choice set
// whose link-path incidence matrix is loaded into link variables
// of the form (interactions with distance or free-flow time)
// Pjj_DIST, Pjj_DIST1, Pjj_FFT, Pjj_FFT1 where jj is the route
// number and the xxxxxx1 variables indicate traversing the link
// in the direction opposite from the one in which the link was
// digitized.
//
```

```
// The parameters passed in the array ``args'' are:
// 1. defn (integer): 2=current Li/Lj formulation with exponent parameter
//       Currently, only definition 2 is coded
//       In this definition, a two-way road is treated as two one-way links
//
//       PSi = sum over arcs a ( la / Li * LSCai )
//
//       LSCai = 1 if a is a centroid connector
//
//             = 1 / sum over paths j ( Li / Lj * delta aj )^gamma
//
//             = 1/ Li^gamma sum over j ( 1/ Lj * delta aj )^gamma
//
//       where delta aj = 1 if path j uses arc a, 0 o/w
//
//       for real links, LSCai = [ 1/ sum over j ( 1/ Lj * delta aj )^gamma ]
/ Li^gamma
//
// 2. gamma (real): gamma for definition 2
// 3. lkv_name (string): name of link view
// 4. lks_name (string): set name with links used (to reduce run time)
// 5. tot_k (integer): number of paths produced
// 6. skv_name (string): name of skim view (where path sizes will be stored)
// 7. obsno (integer): observation number (to find record in skim table)
// 8. flag (integer): if 1, calculate only distance-based path sizes
//                    if 2, calculate only free-flow-time-based path sizes
//                    if 3, calculate both
//                    higher powers of two reserved for
//                    other impedance variables
// ----------------------------------------------------------------

// start with some defaults
// ========================

defn=2
gamma=4
lkv_name="New Mystic K52 Data"
lks_name="tmp_used"
tot_k=52
skv_name="New Mystic Skim Table"
obsno=2
flag=3

dim path_dsts[52]
dim path_ffts[52]
dim psd[52]
dim psf[52]

temp_dir="d:\\users\\sramming\\tc\\tmp3\\"

num_args=ArrayLength(args)

if num_args>=1 then defn    =args[1]
if num_args>=2 then gamma   =args[2]
if num_args>=3 then lkv_name=args[3]
if num_args>=4 then lks_name=args[4]
if num_args>=5 then tot_k   =args[5]
if num_args>=6 then skv_name=args[6]
if num_args>=7 then obsno   =args[7]
if num_args>=8 then flag    =args[8]

dist_based=Mod(flag,2)=1
fft_based=Floor(Mod(flag,4)/2)

// assemble path impedances
// ========================

if dist_based=1 then do
    // assemble distances
    // ==================
    for jj=1 to tot_k do
        if jj=1 then do
            imp_arr={ {"p1_dist","sum"}, {"p1_dist1","sum"} }
            sum_flds={"SUMP1_DIST","SUMP1_DIS1"}
        end
        else do // add another
            ins_arr={ {"p"+String(jj)+"_dist","sum"},
                      {"p"+String(jj)+"_dist1","sum"} }
            imp_arr=InsertArrayElements(imp_arr,jj*2,ins_arr)
            kw=1
            if jj>9 then kw=2
            ins_arr={"SUMP"+String(jj)+Substring("_DIST",1,6-kw),
                     "SUMP"+String(jj)+Substring("_DIS",1,5-kw)+"1"}
            sum_flds=InsertArrayElements(sum_flds,jj*2,ins_arr)
        end
    end // loop to create skim flags

    tmp_view="r"+String(obsno)+"psd"
    tmp_fnam=temp_dir+tmp_view+".dbf"

    AggregateTable(tmp_view,lkv_name+"|tmp_used","dBASE",
          tmp_fnam,"ONE",imp_arr, null )

    tmp_rh=GetFirstRecord(tmp_view+"|",null)
    // first and only record
    tmp_dsts=GetRecordValues(tmp_view,tmp_rh,sum_flds)

    CloseView(tmp_view)

    for jj=1 to tot_k do
        path_dsts[jj]=tmp_dsts[2*jj-1][2]+tmp_dsts[2*jj][2]
    end
end // if distance-based path-sizes requested

if fft_based=1 then do
    // assemble free-flow times
    // ========================
    for jj=1 to tot_k do
        if jj=1 then do
            imp_arr={ {"p1_fft","sum"}, {"p1_fft1","sum"} }
            sum_flds={"SUMP1_FFT","SUMP1_FFT1"}
        end
        else do // add another
            ins_arr={ {"p"+String(jj)+"_fft","sum"},
                      {"p"+String(jj)+"_fft1","sum"} }
            imp_arr=InsertArrayElements(imp_arr,jj*2,ins_arr)
            kw=1
            if jj>9 then kw=2
            ins_arr={"SUMP"+String(jj)+"_FFT",
                     "SUMP"+String(jj)+Substring("_FFT",1,5-kw)+"1"}
            sum_flds=InsertArrayElements(sum_flds,jj*2,ins_arr)
        end
    end // loop to create skim flags

    tmp_view="r"+String(obsno)+"psf"
    tmp_fnam=temp_dir+tmp_view+".dbf"

    AggregateTable(tmp_view,lkv_name+"|tmp_used","dBASE",
          tmp_fnam,"ONE",imp_arr, null )

    tmp_rh=GetFirstRecord(tmp_view+"|",null)
    // first and only record
    tmp_ffts=GetRecordValues(tmp_view,tmp_rh,sum_flds)

    CloseView(tmp_view)

    for jj=1 to tot_k do
        path_ffts[jj]=tmp_ffts[2*jj-1][2]+tmp_ffts[2*jj][2]
    end
end // if free-flow-time-based path-sizes requested

// initialize path size arrays
// ==========================
for jj=1 to tot_k do
    if dist_based=1 then psd[jj]=0
    if fft_based=1  then psf[jj]=0
end

// prepare to loop over links
// ==========================
SetView(lkv_name)
tot_lks=GetSetCount(lks_name)
did_lks=0
last_pct=0

// create array of variable names for extracting LP interactions
// =============================================================
if dist_based=1 then do
    for jj=1 to tot_k do
        if jj=1 then do
            dist_flds={"P1_DIST","P1_DIST1"}
        end
        else do // add another
            ins_arr={ "P"+String(jj)+"_DIST", "p"+String(jj)+"_DIST1" }
            dist_flds=InsertArrayElements(dist_flds,jj*2-1,ins_arr)
        end
    end // loop to create skim flags
end // if distance based
if fft_based=1  then do
    for jj=1 to tot_k do
        if jj=1 then do
```

```
            fft_flds={"P1_FFT","P1_FFT1"}
        end
        else do // add another
            ins_arr={ "P"+String(jj)+"_FFT", "p"+String(jj)+"_FFT1" }
            fft_flds=InsertArrayElements(fft_flds,jj*2-1,ins_arr)
        end
    end // loop to create skim flags
end // if free-flow based

// begin loop over (used) links
// ===========================
CreateProgressBar("Calculating Path Sizes","False")
lk_rh=GetFirstRecord(lkv_name+"|"+lks_name,null)
while (lk_rh<>null) and (did_lks<tot_lks) do
    SetRecord(null,lk_rh)

    // get LP incidence arrays
    // ======================
    if dist_based=1 then do
        dist_lpi=GetRecordValues(lkv_name,lk_rh,dist_flds)
        link_dist=lkv_name.DISTANCE
    end
    if fft_based=1 then do
        fft_lpi =GetRecordValues(lkv_name,lk_rh,fft_flds)
        link_fft=lkv_name.FF_TIME
    end

    // for each link, we calculate length of arc * LSCai
    //
    // at the end, we divide by Li
    // ================================================
    // all this is PS definition 2
    // currently no if/select branch for it or other defns

    if lkv_name.CENT_CONN=1 then do // centroid connector
        for jj=1 to tot_k do
            if dist_based=1 then do
                if dist_lpi[jj*2-1][2]+dist_lpi[jj*2][2]>0 then
                    psd[jj]=psd[jj]+link_dist
            end
            if fft_based=1 then do
                if fft_lpi[jj*2-1][2]+fft_lpi[jj*2][2]>0 then
                    psf[jj]=psf[jj]+link_fft
            end
        end
    end // give LSCs of one to all paths
    else do // real link
        max_lk_paths=lkv_name.KS_USED
        // not necessarily useful because some links have 0 impedance
        // plus paths may go forward or backwards

        // allocate link size contributions for forward (0) direction
        // ==========================================================
        // count paths
        done=0
        first_k=1
        last_k=tot_k
        fwd_path_cnt=0
        jj=1
        while (done=0) and (jj<=tot_k) do
            if dist_based=1 then do
                if dist_lpi[jj*2-1][2]>0 then do
                    fwd_path_cnt=fwd_path_cnt+1
                    last_k=jj
                    if fwd_path_cnt=1 then first_k=jj
                    if fwd_path_cnt=max_lk_paths then done=1
                end
            end // if dist-based
            else if fft_based=1 then do
                if fft_lpi[jj*2-1][2]>0 then do
                    fwd_path_cnt=fwd_path_cnt+1
                    last_k=jj
                    if fwd_path_cnt=1 then first_k=jj
                    if fwd_path_cnt=max_lk_paths then done=1
                end
            end // if fft-based
            jj=jj+1
        end // while loop over paths
        // what I take from this loop is fwd_path_cnt, first_k and last_k

        if fwd_path_cnt=1 then do // easy, since LSC is one
            if dist_based=1 then psd[first_k]=psd[first_k]+link_dist
            if fft_based=1  then psf[first_k]=psf[first_k]+link_fft
        end
        else if fwd_path_cnt>1 then do // allocate LSC
// ------------------------------------------------------------------
// recall that for real links
//
//       LSCai = 1 / sum over paths j ( Li / Lj * delta aj )^gamma
//
//       LSCai = sum over j ( Lj * delta aj )^gamma / Li^gamma
// ------------------------------------------------------------------
            // loop to calculate LSC path sum
            // ------------------------------
            if dist_based=1 then dist_psum=0
            if fft_based=1  then fft_psum=0
            for jj=first_k to last_k do
                if dist_based=1 then
                    if dist_lpi[jj*2-1][2]>0
                    then dist_psum=dist_psum
                           +1/Pow(path_dsts[jj],gamma)
                // hope TransCAD has enough internal precision
                if fft_based=1 then
                    if fft_lpi[jj*2-1][2]>0
                    then fft_psum=fft_psum
                            +1/Pow(path_ffts[jj],gamma)
            end // jj loop
            // -------------------------------
            // loop to add allocated LSCs
            // -------------------------------
            for jj=first_k to last_k do
                if dist_based=1 then
                    if dist_lpi[jj*2-1][2]>0 then psd[jj]=psd[jj]
                        +link_dist/(Pow(path_dsts[jj],gamma)*dist_psum)
                if fft_based=1 then
                    if fft_lpi[jj*2-1][2]>0 then psf[jj]=psf[jj]
                        +link_fft/(Pow(path_ffts[jj],gamma)*dist_psum)
            end // jj loop
        end // else if more than one path uses the link
        // else it's zero, so I don't need to add anything to path sizes

        // allocate link size contributions for reverse (1) direction
        // ==========================================================
        max_bw_paths=max_lk_paths-fwd_path_cnt

        done=0
        first_k=1
        last_k=tot_k
        bkw_path_cnt=0
        jj=1
        while (done=0) and (jj<=tot_k) do
            if dist_based=1 then do
                if dist_lpi[jj*2][2]>0 then do
                    bkw_path_cnt=bkw_path_cnt+1
                    last_k=jj
                    if bkw_path_cnt=1 then first_k=jj
                    if bkw_path_cnt=max_bw_paths then done=1
                end
            end // if dist-based
            else if fft_based=1 then do
                if fft_lpi[jj*2][2]>0 then do
                    bkw_path_cnt=bkw_path_cnt+1
                    if bkw_path_cnt=1 then first_k=jj
                    if bkw_path_cnt=max_bw_paths then done=1
                end
            end // if fft-based
            jj=jj+1
        end // while loop over paths
        // what I take from this loop is fwd_path_cnt, first_k and last_k

        if bkw_path_cnt=1 then do // easy, since LSC is one
            if dist_based=1 then psd[first_k]=psd[first_k]+link_dist
            if fft_based=1  then psf[first_k]=psf[first_k]+link_fft
        end
        else if bkw_path_cnt>1 then do // allocate LSC

            // loop to calculate LSC numerator
            // -------------------------------
            if dist_based=1 then dist_psum=0
            if fft_based=1  then fft_psum=0
            for jj=first_k to last_k do
                if dist_based=1 then
                    if dist_lpi[jj*2][2]>0
                    then dist_psum=dist_psum
                              +1/Pow(path_dsts[jj],gamma)
                if fft_based=1 then
                    if fft_lpi[jj*2][2]>0
                    then fft_psum=fft_psum
                              +1/Pow(path_ffts[jj],gamma)
            end // jj loop
            // --------------------------------
```

```
            // loop to add allocated LSCs
            // ------------------------------
            for jj=first_k to last_k do
                if dist_based=1 then
                    if dist_lpi[jj*2][2]>0 then psd[jj]=psd[jj]
                        +link_dist/(Pow(path_dsts[jj],gamma)*dist_psum)
                if fft_based=1 then
                    if fft_lpi[jj*2][2]>0 then psf[jj]=psf[jj]
                        +link_fft/(Pow(path_ffts[jj],gamma)*fft_psum)
            end // jj loop
        end // else if more than one path uses the link
        // else it's zero, so I don't need to add anything to path sizes

    end // allocate LSCs among real links

    did_lks=did_lks+1
    curr_pct=Floor(100*did_lks/tot_lks)
    if curr_pct>last_pct then do
        UpdateProgressBar("Calculating Path Sizes ("+String(did_lks)+" of "
                          +String(tot_lks)+" links)",curr_pct)
        last_pct=curr_pct
    end
    lk_rh=GetNextRecord(lkv_name+"|"+lks_name,null,null)
end // loop over links
DestroyProgressBar()

// now normalize by path impedances
// ================================
for jj=1 to tot_k do
    if dist_based=1 then psd[jj]=psd[jj]/path_dsts[jj]
    if fft_based=1  then psf[jj]=psf[jj]/path_ffts[jj]
end

// copy to skim view
// =================
SetView(skv_name)
obsskmrh=LocateRecord(skv_name+"|","OBSNO",{obsno},{"Exact","True"})
SetRecord(null,obsskmrh)

skv_name.PS_DEF=defn
skv_name.PS_PARAM=gamma

if dist_based=1 then do
    // assemble set record value arrays
    // ================================
    for jj=1 to tot_k do
        if jj=1 then skim_put={ {"PS1DIST",psd[1]} }
        else do
            ins_arr={ {"PS"+String(jj)+"DIST",psd[jj]} }
            skim_put=InsertArrayElements(skim_put,jj,ins_arr)
        end
    end // path loop
    SetRecordValues(skv_name,obsskmrh,skim_put)
end // dist-based

if fft_based=1 then do
    // assemble set record value arrays
    // ================================
    for jj=1 to tot_k do
        if jj=1 then skim_put={ {"PS1FFT",psf[1]} }
        else do
            ins_arr={ {"PS"+String(jj)+"FFT",psf[jj]} }
            skim_put=InsertArrayElements(skim_put,jj,ins_arr)
        end
    end // path loop
    SetRecordValues(skv_name,obsskmrh,skim_put)
end // fft-based

endMacro
```

C.5 Macro to Calculate Path Sizes from LPI Matrix

```
/////////////////////////////////////////////////////////
// file://magellancee.mit.edu/d:/users/sramming/tc_mac/quik_ps.rsc
//
// Scott Ramming
// M 26 Feb 2001
//
// Write the LP incidence matrix for an observation or set
//
// ADDITION: R 29 March 2001 - PS defn 3 = ``Winners-take-all'',
// ``Losers-ignored'' or All-Shortest-Paths PS
//
// If Li is within ps_gamma of L*a = min over d aj = 1 of Lj
// Then path i shares LSC equally among other shortest paths
//
// ADDITION: M 10 April 2001 - PS defn 4 = ``harmonic''
//
// Moshe's definition of W 4 April 2001 committee meeting
//
// LSC ain = 1 / sum over j of 1( Lj <= Li + ps_gamma )
//
// like PS defn 3, ps_gamma is a tolerance here
//
// Note: in this defn, sum over j of LSC ajn >= 1
//
// ``harmonic'' name comes from sorting LSC by ascending Li
//
//////////////////////////////////////////////////////////

Macro "Quick PS"

// --------------------------------------------------------------------------
// Run-related variables:

max_k=51 // the most spaces in the skim and link-path files

slp_fn="selectio.csv"
// name of file with link-path incidence (LPI) matrix

ps_defn=3
ps_gamma=0.01

dst_based=1
fft_based=1

// --------------------------------------------------------------------------
// Project-related variables:

temp_dir ="d:\\users\\sramming\\tc\\tmp3\\"
print_dir="d:\\users\\sramming\\tc\\newmystc\\"
link_dir ="d:\\users\\sramming\\tc\\newmystc\\paths\\"

lk_view="New Mystic K52 Data"
```

```
skm_view="New Mystic Skim Table"

// ---------------------------------------------------------
// internal variables

err_lev=0

dim lkid_arr[900]  // holds id and dir of link
dim lp_arr[900]    // holds a string of 51 zeros and ones
dim dst_arr[900]   // holds link distance
dim fft_arr[900]   // holds link free-flow time
dim jfst_arr[900]  // holds index of first path using link
dim jlst_arr[900]  // holds index of last path using link
dim na_arr[900]    // holds Na = number of paths using link
                   //         = sum over j delta aj
                   //         = number of ones in lp_arr[a]
dim cc_arr[900]    // 1 if link a is a centroid connector, 0 otherwise

dim path_dsts[52]  // holds Li for i = 1 to 52
dim path_ffts[52]
dim psd[52]
dim psf[52]
dim sps_dst[52] // array of shortest paths (based on dist)
                // using link for PS defn 3
                // use with num_spsd
dim sps_fft[52] // similar array for fastest paths
dim pth_idx[52] // temporary array of path numbers used by current link
                // in PS defn 4 -- for speed-up

read_lks=R2I(0)

new_lp_line="100000000000000000000000000000000000000000000000000"
// one 1 followed by 50 zeroes

// ------------------------------------------------------------------
// open LP matrix file and read first observation header
// =====================================================
// MultiLabelCSG 1.0                          // version ident
// rt_set,tot_rts,tot_lbls,max_k              // run params
// label_array                                // list of labels
// route_name,orig_rh,dest_rh                 // route header
// found_k,curr_lbl,label_array,num_links     // unique path header
// link ID, direction flag                    // link info
// [repeats]
// -1                                         // verify end of link info
// [repeat from unique path header]
// -9                                         // end of unique routes
//                                       (total number unknown beforehand)
// min(found_k,max_k),k of best match,best match overlap
                                              // path gen summary for route
// -19                                        // verify end of obs info
// -39                                        // verify end of all obs and
                                              // therefore of file

SetView(lk_view)

lnk_file=OpenFile(link_dir+slp_fn,"r")
```

```
lk_temp=ReadLine(lnk_file)
// first should be header
lk_temp=ReadLine(lnk_file)
od_info=ParseString(lk_temp,",")
// checks
tot_obs=S2I(od_info[2])
tot_lbls=S2I(od_info[3])

for i=1 to tot_lbls do
// read through label array
    lk_temp=ReadLine(lnk_file)
end

did_obs=0

EnableProgressBar("Status ...",3)
CreateProgressBar("Reading 1st of "+String(tot_obs)+" observations.","True")

while (did_obs<tot_obs) and (err_lev<8) do // giant loop over obs
    lk_temp=ReadLine(lnk_file)
    // name, orig and dest
    rt_info=ParseString(lk_temp,",")
    if S2I(rt_info[1]) = -39 then do
       did_obs=tot_obs
    end
    else if ArrayLength(rt_info)<>3 then do
        ShowMessage("Unexpected route header.")
        RunDBox("G30 Show Array","Route Info",rt_info)
        obsno=3904 // harmless obs to dump gibberish on
    end
    else do // big survey route (obs) loop -- no indent
    did_obs=did_obs+1
    rt_name=rt_info[1]
    obsno=S2I(Substring(rt_name,3,4))

    read_lks=R2I(0)
    new_lp_line="100000000000000000000000000000000000000000000000000"
    // one 1 followed by 50 zeroes
    did_rts=0
    fin_w_od=0

    CreateProgressBar("Reading first route.","True")
    UpdateProgressBar("Reading first route.",0)

    // read in routes
    // =============
    while (did_rts<=max_k) and (fin_w_od=0) do // loop over alt routes
        did_rts=did_rts+1

        if dst_based=1 then path_dsts[did_rts]=0
        if fft_based=1 then path_ffts[did_rts]=0

        lk_temp=ReadLine(lnk_file)
        rt_info=ParseString(lk_temp,",")
        if ArrayLength(rt_info)<>4 then do
            // check that we're not done with routes
            if S2I(rt_info[1]) = -9 then do
                fin_w_od=1
                did_rts=did_rts-1 // to undo the increment above
            end
            else if S2I(rt_info[1]) = -39 then do
                fin_w_od=1
                did_rts=did_rts-1 // to undo the increment above
                did_obs=tot_obs
            end
            else do
                // something's wrong
                ShowMessage("Unexpected Route Info.")
                fin_w_od=1
                did_rts=did_rts-1 // to undo the increment above
                RunDBox("G30 Show Array","Route Info Array",rt_info)
            end
        end
        else do // found 4 records -- no indent
        ck_rts=S2I(rt_info[1])
        num_lks=S2I(rt_info[4])

        if did_rts<>ck_rts then do
            // shouldn't get here
            ShowMessage("Did_rts = "+String(did_rts)+" and ck_rts = "
                +String(ck_rts) )
        end
        else do
        // read links for each route and put in LP matrix
        // ==============================================
            did_lks=0
            fin_w_lks=0

            while(did_lks<=num_lks) and (fin_w_lks=0) do
                lk_temp=ReadLine(lnk_file)
                lk_info=ParseString(lk_temp,",")
                if ArrayLength(lk_info)<>2 then do
                // something might be wrong
                    if S2I(lk_info[1])=-1 then fin_w_lks=1
                    // things terminated OK
                    else do
                        ShowMessage("Unexpected number of link descriptors")
                        RunDBox("G30 Show Array","Current Line",lk_info)
                        fin_w_lks=1
                    end
                end
                else do // found link OK
                    did_lks=did_lks+1
                    // now there are two cases
                    // we've seen the link ID before or it's new
                    // -----------------------------------------
                    new_link=0
                    if read_lks=0 then new_link=1
                    else do
                        test=Subarray(lkid_arr,1,read_lks)
                        lk_ar_pos=R2I(ArrayPosition(test,{lk_temp},null))
                        if lk_ar_pos=0 then new_link=1
                    end
```

```
if new_link=1 then do
    read_lks=R2I(read_lks+1)
    lkid_arr[read_lks]=lk_temp
    lp_arr[read_lks]=new_lp_line
    jfst_arr[read_lks]=did_rts
    jlst_arr[read_lks]=did_rts
    na_arr[read_lks]=1

    lk_id= S2I(lk_info[1])

    lk_rh=ID2RH(lk_id)
    SetRecord(lk_view,lk_rh)
    cc_arr[read_lks]=lk_view.CENT_CONN
    if dst_based=1 then do
        dst_arr[read_lks]=lk_view.DISTANCE
        path_dsts[did_rts]=path_dsts[did_rts]
            +dst_arr[read_lks]
    end
    if fft_based=1 then do
        fft_arr[read_lks]=lk_view.FF_TIME
        path_ffts[did_rts]=path_ffts[did_rts]
            +fft_arr[read_lks]
    end
end // if new link

else do // repeated link
        // ============
    if did_rts=1 then do
        // shouldn't happen -- maybe a cycle?
        lp_arr[lk_ar_pos]="1"
            +Substring(lp_arr[lk_ar_pos],2,max_k-1)
        jfst_arr[lk_ar_pos]=1
        jlst_arr[lk_ar_pos]=1

        if dst_based=1 then do
            path_dsts[did_rts]=path_dsts[did_rts]
                +dst_arr[lk_ar_pos]
        end
        if fft_based=1 then do
            path_ffts[did_rts]=path_ffts[did_rts]
                +fft_arr[lk_ar_pos]
        end
    end // if first route and repeated link

    else if did_rts=max_k then do
        lp_arr[lk_ar_pos]=Substring(lp_arr[lk_ar_pos],1,
            max_k-1)+"1"
        jlst_arr[lk_ar_pos]=max_k
        na_arr[lk_ar_pos]=na_arr[lk_ar_pos]+1

        if dst_based=1 then do
            path_dsts[did_rts]=path_dsts[did_rts]
                +dst_arr[lk_ar_pos]
        end
        if fft_based=1 then do
            path_ffts[did_rts]=path_ffts[did_rts]
```

```
                                +fft_arr[lk_ar_pos]
                        end
                    end // if last route

                    else do
                        lp_arr[lk_ar_pos]=Substring(lp_arr[lk_ar_pos],
                            1,did_rts-1)+"1"+Substring(lp_arr[lk_ar_pos],
                            did_rts+1,51-did_rts)
                        jlst_arr[lk_ar_pos]=did_rts
                        na_arr[lk_ar_pos]=na_arr[lk_ar_pos]+1

                        if dst_based=1 then do
                            path_dsts[did_rts]=path_dsts[did_rts]
                                +dst_arr[lk_ar_pos]
                        end
                        if fft_based=1 then do
                            path_ffts[did_rts]=path_ffts[did_rts]
                                +fft_arr[lk_ar_pos]
                        end
                    end
                end // repeated link

            end // else found link ID OK
        end // while reading links loop
    end // alt route numbers (did_rts and ck_rts) matched (else branch)
    end // found 4 records for a legitimate alt route (else branch)
    // may need to move lower

    usr_tmp=UpdateProgressBar("Read "+String(did_rts)+" routes.",
        Floor(100*did_rts/max_k) )

    new_lp_line="0"+Substring(new_lp_line,1,50)

    if usr_tmp="True" then do
        fin_w_od=1
        err_lev=39
    end
end // while loop over alt routes

DestroyProgressBar()

// did_rts is now a useful value
// in most cases, did_rts should be preferred over max_k now

rt_temp=ReadLine(lnk_file)
if Substring(rt_temp,1,1)="-" then rt_temp=ReadLine(lnk_file)
if Substring(rt_temp,1,1)="-" then rt_temp=ReadLine(lnk_file)
// may be a -1 or -9 to throw away
// three summaries of CSG for OD
rt_info=ParseString(rt_temp,",")
if S2I(rt_info[1])<>did_rts then do // something goofy
    if S2I(rt_info[1])=-19 then do // just got here early
    end
    else do
        ShowMessage("Unexpected CSG Summary Info.")
        RunDBox("G30 Show Array","CSG Summary",rt_info)
        err_lev=7
    end
end
// else do
rt_temp=ReadLine(lnk_file)
// throw away the -19
// end

if err_lev<7 then do // not indented
// Now can do route processing for OD
// ==================================
// in this case, calculate path sizes

// initialize path size arrays
// ===========================
for jj=1 to max_k do
    if dst_based=1 then psd[jj]=0
    if fft_based=1 then psf[jj]=0
end

// prepare to loop over links
// ==========================
last_pct=0

// begin loop over (used) links
// ============================
CreateProgressBar("Calculating Path Sizes","True")
for aa = 1 to read_lks do

    // LP incidence arrays are in lp_arr
    // =================================

    // for each link, we calculate length of arc * LSCai
    //
    // at the end, we divide by Li
    // =================================================
    // all this is PS definition 0 and 2

    if cc_arr[aa]>=1 then do // centroid connector
                             // ==================
        for jj=jfst_arr[aa] to jlst_arr[aa] do
            if Substring(lp_arr[aa],jj,1)="1" then do
                if dst_based=1 then psd[jj]=psd[jj]+dst_arr[aa]
                if fft_based=1 then psf[jj]=psf[jj]+fft_arr[aa]
            end
        end
    end // give LSCs of one to all paths

    else do // real link
            // ===================
        // na_arr is reliable here

        // allocate link size contributions
        // ==========================================================
        // I have path_cnt, first_k and last_k in arrays
        //        na_arr  , jfst_arr,   jlst_arr

        if na_arr[aa]=1 then do // easy, since LSC is one
```

```
            // also true for defn 0 : Na = 1; LSC = 1/Na = 1
                if dst_based=1 then psd[jfst_arr[aa]]=psd[jfst_arr[aa]]
                    +dst_arr[aa]
                if fft_based=1 then psf[jfst_arr[aa]]=psf[jfst_arr[aa]]
                    +fft_arr[aa]
            end
            else if na_arr[aa]>1 then do // allocate LSC

                if ps_defn=0 then do
                    for jj=jfst_arr[aa] to jlst_arr[aa] do
                        if Substring(lp_arr[aa],jj,1)="1" then do
                            if dst_based=1 then psd[jj]=psd[jj]
                                +dst_arr[aa]/na_arr[aa]
                            if fft_based=1 then psf[jj]=psf[jj]
                                +fft_arr[aa]/na_arr[aa]
                        end
                    end
                end // if defn=0

                else if ps_defn=2 then do
// ------------------------------------------------------------------
// recall that for real links
//
//       LSCai = 1 / sum over paths j ( Li / Lj * delta aj )^gamma
//
//       LSCai = sum over j ( Lj * delta aj )^gamma / Li^gamma
// ------------------------------------------------------------------
                    // loop to calculate LSC path sum
                    // -------------------------------
                    if dst_based=1 then dst_psum=0
                    if fft_based=1 then fft_psum=0
                    for jj=jfst_arr[aa] to jlst_arr[aa] do
                        if Substring(lp_arr[aa],jj,1)="1" then do
                            if dst_based=1 then dst_psum=dst_psum
                                +1/Pow(path_dsts[jj],ps_gamma)
                            if fft_based=1 then fft_psum=fft_psum
                                +1/Pow(path_ffts[jj],ps_gamma)
                        end
                    end // jj loop
                    // --------------------------------
                    // loop to add allocated LSCs
                    // --------------------------------
                    for jj=jfst_arr[aa] to jlst_arr[aa] do
                        if Substring(lp_arr[aa],jj,1)="1" then do
                            if dst_based=1 then psd[jj]=psd[jj]
                                +dst_arr[aa]/
                                 (Pow(path_dsts[jj],ps_gamma)*dst_psum)
                            if fft_based=1 then psf[jj]=psf[jj]
                                +fft_arr[aa]/
                                 (Pow(path_ffts[jj],ps_gamma)*fft_psum)
                        end
                    end // jj loop
                end // defn = 2

                else if ps_defn=3 then do
                    // --------------------------------------
                    // first find the minimum impedance paths
```

```
    // --------------------------------------
    if dst_based=1 then lstara_d=9999
    if fft_based=1 then lstara_f=9999
    for jj=jfst_arr[aa] to jlst_arr[aa] do
        if Substring(lp_arr[aa],jj,1)="1" then do
            if dst_based=1 then
                lstara_d=Min(lstara_d,path_dsts[jj])
            if fft_based=1 then
                lstara_f=Min(lstara_f,path_ffts[jj])
        end
    end // jj loop
    // --------------------------------------
    // now count paths qualifying as shortest
    // --------------------------------------
    if dst_based=1 then num_spsd=0
    // because we'll find it later
    if fft_based=1 then num_spsf=0
    for jj=jfst_arr[aa] to jlst_arr[aa] do
        if Substring(lp_arr[aa],jj,1)="1" then do
            if dst_based=1 then
                if path_dsts[jj]-lstara_d<ps_gamma then do
                    num_spsd=num_spsd+1
                    sps_dst[num_spsd]=jj
                end // if a short path
            if fft_based=1 then
                if path_ffts[jj]-lstara_f<ps_gamma then do
                    num_spsf=num_spsf+1
                    sps_fft[num_spsf]=jj
                end // if a quick path
        end // if path jj uses link aa
    end // jj loop
    // --------------------------------------
    // finally, loop to allocate LSCs
    // --------------------------------------
    if dst_based=1 then do
        for jj=1 to num_spsd do
            psd[sps_dst[jj]]=psd[sps_dst[jj]]
                +dst_arr[aa]/num_spsd
        end // jj loop over short paths
    end // if dst_based
    if fft_based=1 then do
        for jj=1 to num_spsf do
            psf[sps_fft[jj]]=psf[sps_fft[jj]]
                +fft_arr[aa]/num_spsf
        end // jj loop over quick paths
    end // if fft_based
end // if ps_defn=3

// PS defn 4 = Harmonic Path Size
// proposed by Moshe W 4 April 2001
// ====================================
else if ps_defn=4 then do
    // first, put path indices (that is, a number 1 to 51)
    // in the pth_ind array so future steps will be faster
    // ---------------------------------------------------
    curr_idx=1
    for jj=jfst_arr[aa] to jlst_arr[aa] do
```

```
                if Substring(lp_arr[aa],jj,1)="1" then do
                    pth_idx[curr_idx]=jj
                    curr_idx=curr_idx+1
                end // if path uses link
            end // jj loop
            // now curr_idx should be 1 more than na_arr[aa]

            // in this part, we loop (jj) over
            // all paths using link aa
            // to calculate the denominator of their LSC;
            // this requires a second internal loop (kk)
            // to compare path impedances
            // --------------------------------------------------
            for jj=1 to na_arr[aa] do
                if dst_based=1 then num_spsd=0
                // because we'll find it later
                if fft_based=1 then num_spsf=0
                for kk=1 to na_arr[aa] do
                    if dst_based=1 then do
                        if path_dsts[pth_idx[kk]]<=
                          (path_dsts[pth_idx[jj]] + ps_gamma)
                        then num_spsd=num_spsd+1
                    end // if dist based
                    if fft_based=1 then do
                        if path_ffts[pth_idx[kk]]<=
                          (path_ffts[pth_idx[jj]] + ps_gamma)
                        then num_spsf=num_spsf+1
                    end // if fft based
                end // kk loop

                // add path jj's LSC
                // -----------------
                if dst_based=1 then do
                    psd[pth_idx[jj]]=psd[pth_idx[jj]]
                      +dst_arr[aa]/num_spsd
                end // if dst_based
                if fft_based=1 then do
                    psf[pth_idx[jj]]=psf[pth_idx[jj]]
                      +fft_arr[aa]/num_spsf
                end // if fft_based
            end // jj loop
        end // if ps_defn=4

    end // else if more than one path uses the link
    // else it's zero, so I don't need to add anything to path sizes

end // else it's a real link

curr_pct=Floor(100*aa/read_lks)
if curr_pct>last_pct then do
    usr_tmp=UpdateProgressBar("Calculating Path Sizes ("
                  +String(aa)+" of "
                  +String(read_lks)+" links)",curr_pct)
    last_pct=curr_pct

    if usr_tmp="True" then do
        aa=read_lks
```

```
                err_lev=37
            end
        end
    end // PS aa loop over links
    DestroyProgressBar()

    // now normalize by path impedances
    // ================================
    for jj=1 to did_rts do
        if dst_based=1 then psd[jj]=psd[jj]/path_dsts[jj]
        if fft_based=1 then psf[jj]=psf[jj]/path_ffts[jj]
    end

    // copy to skim view
    // =================
    SetView(skm_view)
    obsskmrh=LocateRecord(skm_view+"|","OBSNO",{obsno},
        {"Exact","True"})
    SetRecord(null,obsskmrh)

    skm_view.PS_DEF=ps_defn
    skm_view.PS_PARAM=ps_gamma

    if dst_based=1 then do
        // assemble set record value arrays
        // ================================
        for jj=1 to max_k do
        // with max_k instead of did_rts,
        // we'll have zeros as size of ungenerated (unavailable) paths
            if jj=1 then skim_put={ {"PS1DIST",psd[1]} }
            else do
                ins_arr={ {"PS"+String(jj)+"DIST",psd[jj]} }
                skim_put=InsertArrayElements(skim_put,jj,ins_arr)
            end
        end // for jj path loop

        ins_arr={ {"MAXPSDIST",ArrayMax(Subarray(psd,1,did_rts))} }
        skim_put=InsertArrayElements(skim_put,max_k+1,ins_arr)

        SetRecordValues(skm_view,obsskmrh,skim_put)
    end // if dist-based

    if fft_based=1 then do
        // assemble set record value arrays
        // ================================
        for jj=1 to max_k do
            if jj=1 then skim_put={ {"PS1FFT",psf[1]} }
            else do
                ins_arr={ {"PS"+String(jj)+"FFT",psf[jj]} }
                skim_put=InsertArrayElements(skim_put,jj,ins_arr)
            end
        end // for jj path loop

        ins_arr={ {"MAXPSFFT",ArrayMax(Subarray(psf,1,did_rts))} }
        skim_put=InsertArrayElements(skim_put,max_k+1,ins_arr)

        SetRecordValues(skm_view,obsskmrh,skim_put)
    end // if fft-based

    end // if didn't have an error during the alt-route-reading stage
        // error includes user request for termination

    end // big survey route (obs) loop -- no indent
    // if route header is usable and not -39

    usr_tmp=UpdateProgressBar("Read "+String(did_obs)+" of "
        +String(tot_obs)
        +" observations.",Floor(100*did_obs/tot_obs))

    if usr_tmp="True" then do
        err_lev=39
    end
end // giant loop over observations

DestroyProgressBar()
DisableProgressBar()

ShowMessage("Macro finished with error level "+String(err_lev) )

endMacro
```

C.6 Macro to Calculate Alphas for CNL

```
/////////////////////////////////////////////////////////
// file://magellancee.mit.edu/d:/users/sramming/tc_mac/cnlalpha.rsc
//
// Scott Ramming
// F 9 March 2001
//
// Write the alpha matrix for an observation or set
//////////////////////////////////////////////////////////

Macro "CNL alpha"

// --------------------------------------------------------------------------
// Run-related variables:

imp_var="FF_TIME" // "DISTANCE"
max_k=51 // the most spaces in the skim and link-path files
max_lks=900 // the most links used by an observation

slp_fn="selectio.csv"
// name of the file with the link-path incidence (LPI) data

out_fn="nm_a_fft.txt"

// --------------------------------------------------------------------------
// Project-related variables:

temp_dir ="d:\\users\\sramming\\tc\\tmp3\\"
print_dir="d:\\users\\sramming\\tc\\newmystc\\"
link_dir ="d:\\users\\sramming\\tc\\newmystc\\paths\\"
```

```
lk_view="New Mystic K52 Data"

skm_view="New Mystic Skim Table"

// --------------------------------------------------------
// internal variables

err_lev=0

dim lkid_arr[900]
dim pthimparr[51]
dim alpha_arr[900]

read_lks=R2I(0)

blank_line="0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 "
    +"0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0"
// 51 zeroes with spaces

blank_arr={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
// array of 51 zeros

if imp_var="FF_TIME" then do
    get_parr={"P1FFT"}
    for j=2 to max_k do
        get_parr=InsertArrayElements(get_parr,j,{"P"+String(j)+"FFT"})
    end
end
else if imp_var="DISTANCE" then do
    get_parr={"P1DIST"}
    for j=2 to max_k do
        get_parr=InsertArrayElements(get_parr,j,{"P"+String(j)+"DIST"})
    end
end

// -----------------------------------------------------------------

out_path=print_dir+out_fn
out_fh=OpenFile(out_path,"w")

// open LP matrix file and read first observation header
// =====================================================
// MultiLabelCSG 1.0                         // version ident
// rt_set,tot_rts,tot_lbls,max_k             // run params
// label_array                               // list of labels
// route_name,orig_rh,dest_rh                // route header
// found_k,curr_lbl,label_array,num_links    // unique path header
// link ID, direction flag                   // link info
// [repeats]
// -1                                        // verify end of link info
// [repeat from unique path header]
// -9                                        // end of unique routes
//                                       (total number unknown beforehand)
// min(found_k,max_k),k of best match,best match overlap
                                             // path gen summary for route
// -19                                       // verify end of obs info
// -39                                          // verify end of all obs and
                                                // therefore of file

SetView(lk_view)

lnk_file=OpenFile(link_dir+slp_fn,"r")

lk_temp=ReadLine(lnk_file)
// first should be header
lk_temp=ReadLine(lnk_file)
od_info=ParseString(lk_temp,",")
// checks
tot_obs=S2I(od_info[2])
tot_lbls=S2I(od_info[3])

for i=1 to tot_lbls do
// read through label array
    lk_temp=ReadLine(lnk_file)
end

did_obs=0

EnableProgressBar("Status ...",3)
CreateProgressBar("Reading 1st of "+String(tot_obs)+" observations.","False")

while (did_obs<tot_obs) and (err_lev<8) do // giant loop over obs
    lk_temp=ReadLine(lnk_file)
    // name, orig and dest
    rt_info=ParseString(lk_temp,",")
    if S2I(rt_info[1]) = -39 then do
       did_obs=tot_obs
    end
    else if ArrayLength(rt_info)<>3 then do
        ShowMessage("Unexpected route header.")
        RunDBox("G30 Show Array","Route Info",rt_info)
        obsno=3904 // harmless obs to dump gibberish on
    end
    else do // big route loop -- no indent
    did_obs=did_obs+1
    rt_name=rt_info[1]
    obsno=S2I(Substring(rt_name,3,4))

    read_lks=R2I(0)
    did_rts=0
    fin_w_od=0

    CreateProgressBar("Reading first route.","False")
    UpdateProgressBar("Reading first route.",0)

    // Get path impedances
    // ==================
    SetView(skm_view)
    skm_rh=LocateRecord(skm_view+"|","OBSNO",{obsno},{{"Exact","true"}})
    tmpimparr=GetRecordValues(skm_view,skm_rh,get_parr)
    for j=1 to max_k do
        pthimparr[j]=tmpimparr[j][2]
    end
```

```
// read in routes
// ==============
while (did_rts<=max_k) and (fin_w_od=0) do // loop over alt routes
    did_rts=did_rts+1
    lk_temp=ReadLine(lnk_file)
    rt_info=ParseString(lk_temp,",")
    if ArrayLength(rt_info)<>4 then do
        // check that we're not done with routes
        if S2I(rt_info[1]) = -9 then do
            fin_w_od=1
            did_rts=did_rts-1 // to undo the increment above
        end
        else if S2I(rt_info[1]) = -39 then do
            fin_w_od=1
            did_rts=did_rts-1 // to undo the increment above
            did_obs=tot_obs
        end
        else do
            // something's wrong
            ShowMessage("Unexpected Route Info.")
            fin_w_od=1
            did_rts=did_rts-1 // to undo the increment above
            RunDBox("G30 Show Array","Route Info Array",rt_info)
        end
    end
    else do // found 4 records -- no indent
    ck_rts=S2I(rt_info[1])
    num_lks=S2I(rt_info[4])

    if did_rts<>ck_rts then do
        // shouldn't get here
        ShowMessage("Did_rts = "+String(did_rts)+" and ck_rts = "
            +String(ck_rts) )
    end
    else do
    // read links for each route and put in LP matrix
    // ============================================
        did_lks=0
        fin_w_lks=0

        while(did_lks<=num_lks) and (fin_w_lks=0) do
            lk_temp=ReadLine(lnk_file)
            lk_info=ParseString(lk_temp,",")
            if ArrayLength(lk_info)<>2 then do
            // something might be wrong
                if S2I(lk_info[1])=-1 then fin_w_lks=1
                // things terminated OK
                else do
                    ShowMessage("Unexpected number of link descriptors")
                    RunDBox("G30 Show Array","Current Line",lk_info)
                    fin_w_lks=1
                end
            end
            else do // found link OK
                did_lks=did_lks+1
                SetView(lk_view)
                lk_rh=ID2RH(S2I(lk_info[1]))
                SetRecord(lk_view,lk_rh)
                lkimptmp=GetRecordValues(lk_view,lk_rh,{imp_var})
                lk_imp=lkimptmp[1][2]

                // now there are two cases
                // we've seen the link ID before or it's new
                // ---------------------------------------
                new_link=0
                if read_lks=0 then new_link=1
                else do
                    test=Subarray(lkid_arr,1,read_lks)
                    lk_ar_pos=R2I(ArrayPosition(test,{lk_temp},null))
                    if lk_ar_pos=0 then new_link=1
                end

                if new_link=1 then do
                    read_lks=R2I(read_lks+1)
                    lkid_arr[read_lks]=lk_temp
                    alpha_arr[read_lks]={0,0,0,0,0,0,0,0,0,0,
                     0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
                     0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
                    alpha_arr[read_lks][did_rts]=
                      lk_imp/pthimparr[did_rts]
                end
                else do // repeated link
                    alpha_arr[lk_ar_pos][did_rts]=
                      lk_imp/pthimparr[did_rts]
                end

            end // else found link ID OK
        end // read links loop
    end // alt route numbers matched (else branch)
    end // found 4 records for a legitimate alt route (else branch)
    // may need to move lower

    UpdateProgressBar("Read "+String(did_rts)+" routes.",
        Floor(100*did_rts/max_k) )

end // loop over routes

DestroyProgressBar()

// did_rts is now a useful value

rt_temp=ReadLine(lnk_file)
if Substring(rt_temp,1,1)="-" then rt_temp=ReadLine(lnk_file)
if Substring(rt_temp,1,1)="-" then rt_temp=ReadLine(lnk_file)
// may be a -1 or -9 to throw away
// three summaries of CSG for OD
rt_info=ParseString(rt_temp,",")
if S2I(rt_info[1])<>did_rts then do // something goofy
    if S2I(rt_info[1])=-19 then do // just got here early
    end
    // else if S2I(rt_info[1])=-9 then do
    // end
```

```
        else do
            ShowMessage("Unexpected CSG Summary Info.")
            RunDBox("G30 Show Array","CSG Summary",rt_info)
            err_lev=7
        end
    end
    rt_temp=ReadLine(lnk_file)
    // throw away the -19

    // Now write out the alpha matrix
    // ==============================

    for aa=1 to read_lks do
        line_tmp=""
        for j=1 to max_k do
            line_tmp=line_tmp+String(alpha_arr[aa][j])+" "
        end
        WriteLine(out_fh,line_tmp)
    end
    if read_lks<max_lks then do
        for aa=read_lks+1 to max_lks do
            WriteLine(out_fh,blank_line)
        end
    end

    end // big route loop -- no indent
    // if route header is usable and not -39

    UpdateProgressBar("Read "+String(did_obs)+" of "+String(tot_obs)
        +" observations.",Floor(100*did_obs/tot_obs))
end // giant loop over observations

DestroyProgressBar()
DisableProgressBar()

CloseFile(out_fh)
CloseFile(lnk_file)

ShowMessage("Macro finished with error level "+String(err_lev) )

endMacro
```

C.7 Macro to Calculate Covariance Terms for LK

```
/////////////////////////////////////////////////////////
// file://magellancee.mit.edu/d:/users/sramming/tc_mac/desparse.rsc
//
// Scott Ramming
// M 26 Feb 2001
//
// Write the LP incidence matrix for an observation or set
// for use by Denis' LK program
//////////////////////////////////////////////////////////

Macro "Desparse"
```

```
// --------------------------------------------------------------------------
// Run-related variables:

max_k=51 // the most spaces in the skim and link-path files

slp_fn="surv_est.csv"
// name of the link-path incidence (LPI) data file

// --------------------------------------------------------------------------
// Project-related variables:

temp_dir ="d:\\users\\sramming\\tc\\tmp3\\"
print_dir="d:\\users\\sramming\\tc\\final02\\"
link_dir ="d:\\users\\sramming\\tc\\final02\\"

lk_view="K40 Links"

// --------------------------------------------------------
// internal variables

err_lev=0

dim lkid_arr[900]
dim lp_arr[900]
dim imp_arr[900]

read_lks=R2I(0)

new_lp_line="100000000000000000000000000000000000000000000000000"
// one 1 followed by 50 zeroes

// -----------------------------------------------------------------

lp1_fn="lpsqf159.txt" // "lpsqd159.txt"
lp1_path=print_dir+lp1_fn
lp1_fh=OpenFile(lp1_path,"w")

// the above file has the following format:
// the number of links in respondent n's choice set = Mn
// Mn lines consisting of 51 ones or zeros for the LPI matrix,
//      a space, and the square root of the impedance variable
//      for the diagonal matrix

lp2_fn="lpod159.txt"
lp2_path=print_dir+lp2_fn
lp2_fh=OpenFile(lp2_path,"w")
timenow=GetDateAndTime()
WriteLine(lp2_fh,"Sparse LP matrix to Denis LP matrix -- March 2001")
WriteLine(lp2_fh,"")
WriteLine(lp2_fh,"Run started at "+timenow)
WriteLine(lp2_fh,"")
WriteLine(lp2_fh,"obsno,lks_per_od")

// this second file is more human-readable
// and is used to view the distribution of Mn
// among respondents
// open LP matrix file and read first observation header
// =====================================================

// MultiLabelCSG 1.0                             // version ident
// rt_set,tot_rts,tot_lbls,max_k                 // run params
// label_array                                   // list of labels
// route_name,orig_rh,dest_rh                    // route header
// found_k,curr_lbl,label_array,num_links        // unique path header
// link ID, direction flag                       // link info
// [repeats]
// -1                                            // verify end of link info
// [repeat from unique path header]
// -9                                            // end of unique routes
//                                             (total number unknown beforehand)
// min(found_k,max_k),k of best match,best match overlap
                                                 // path gen summary for route
// -19                                           // verify end of obs info
// -39                                           // verify end of all obs and
                                                 // therefore of file

SetView(lk_view)

lnk_file=OpenFile(link_dir+slp_fn,"r")

lk_temp=ReadLine(lnk_file)
// first should be header
lk_temp=ReadLine(lnk_file)
od_info=ParseString(lk_temp,",")
// checks
tot_obs=S2I(od_info[2])
tot_lbls=S2I(od_info[3])

for i=1 to tot_lbls do
// read through label array
    lk_temp=ReadLine(lnk_file)
end

did_obs=0

EnableProgressBar("Status ...",3)
CreateProgressBar("Reading 1st of "+String(tot_obs)+" observations.","False")

while (did_obs<tot_obs) and (err_lev<8) do // giant loop over obs
    lk_temp=ReadLine(lnk_file)
    // name, orig and dest
    rt_info=ParseString(lk_temp,",")
    if S2I(rt_info[1]) = -39 then do
       did_obs=tot_obs
    end
    else if ArrayLength(rt_info)<>3 then do
        ShowMessage("Unexpected route header.")
        RunDBox("G30 Show Array","Route Info",rt_info)
        obsno=3904 // harmless obs to dump gibberish on
    end
    else do // big route loop -- no indent
    did_obs=did_obs+1
    rt_name=rt_info[1]
```

```
obsno=S2I(Substring(rt_name,3,4))

read_lks=R2I(0)
new_lp_line="100000000000000000000000000000000000000000000000000"
// one 1 followed by 50 zeroes
did_rts=0
fin_w_od=0

CreateProgressBar("Reading first route.","False")
UpdateProgressBar("Reading first route.",0)

// read in routes
// ==============
while (did_rts<=max_k) and (fin_w_od=0) do // loop over alt routes
    did_rts=did_rts+1
    lk_temp=ReadLine(lnk_file)
    rt_info=ParseString(lk_temp,",")
    if ArrayLength(rt_info)<>4 then do
        // check that we're not done with routes
        if S2I(rt_info[1]) = -9 then do
            fin_w_od=1
            did_rts=did_rts-1 // to undo the increment above
        end
        else if S2I(rt_info[1]) = -39 then do
            fin_w_od=1
            did_rts=did_rts-1 // to undo the increment above
            did_obs=tot_obs
        end
        else do
            // something's wrong
            ShowMessage("Unexpected Route Info.")
            fin_w_od=1
            did_rts=did_rts-1 // to undo the increment above
            RunDBox("G30 Show Array","Route Info Array",rt_info)
        end
    end
    else do // found 4 records -- no indent
    ck_rts=S2I(rt_info[1])
    num_lks=S2I(rt_info[4])

    if did_rts<>ck_rts then do
        // shouldn't get here
        ShowMessage("Did_rts = "+String(did_rts)+" and ck_rts = "
            +String(ck_rts) )
    end
    else do
    // read links for each route and put in LP matrix
    // ==============================================
        did_lks=0
        fin_w_lks=0

        while(did_lks<=num_lks) and (fin_w_lks=0) do
            lk_temp=ReadLine(lnk_file)
            lk_info=ParseString(lk_temp,",")
            if ArrayLength(lk_info)<>2 then do
            // something might be wrong
                if S2I(lk_info[1])=-1 then fin_w_lks=1
                // things terminated OK
                else do
                    ShowMessage("Unexpected number of link descriptors")
                    RunDBox("G30 Show Array","Current Line",lk_info)
                    fin_w_lks=1
                end
            end
            else do // found link OK
                did_lks=did_lks+1

                lk_rh=ID2RH(S2I(lk_info[1]))
                SetRecord(lk_view,lk_rh)
                imp=lk_view.FF_TIME
                // ********************************
                // *** this is changed as necessary
                // ********************************

                // now there are two cases
                // we've seen the link ID before or it's new
                // -----------------------------------------
                new_link=0
                if read_lks=0 then new_link=1
                else do
                    test=Subarray(lkid_arr,1,read_lks)
                    lk_ar_pos=R2I(ArrayPosition(test,{lk_temp},null))
                    if lk_ar_pos=0 then new_link=1
                end

                if new_link=1 then do
                    read_lks=R2I(read_lks+1)
                    lkid_arr[read_lks]=lk_temp
                    lp_arr[read_lks]=new_lp_line
                    imp_arr[read_lks]=imp
                end
                else do // repeated link
                    if did_rts=1 then do // shouldn't happen
                        lp_arr[lk_ar_pos]="1"
                          +Substring(lp_arr[lk_ar_pos],2,50)
                    end
                    else if did_rts=51 then do
                        lp_arr[lk_ar_pos]=
                          Substring(lp_arr[lk_ar_pos],1,50)+"1"
                    end
                    else do
                        lp_arr[lk_ar_pos]=Substring(lp_arr[lk_ar_pos],
                            1,did_rts-1)+"1"+Substring(lp_arr[lk_ar_pos],
                            did_rts+1,51-did_rts)
                    end
                end

            end // else found link ID OK
        end // read links loop
    end // alt route numbers matched (else branch)
    end // found 4 records for a legitimate alt route (else branch)
    // may need to move lower

    UpdateProgressBar("Read "+String(did_rts)+" routes.",
            Floor(100*did_rts/max_k) )

        new_lp_line="0"+Substring(new_lp_line,1,50)
    end // loop over routes

    DestroyProgressBar()

    // did_rts is now a useful value

    rt_temp=ReadLine(lnk_file)
    if Substring(rt_temp,1,1)="-" then rt_temp=ReadLine(lnk_file)
    if Substring(rt_temp,1,1)="-" then rt_temp=ReadLine(lnk_file)
    // may be a -1 or -9 to throw away
    // three summaries of CSG for OD
    rt_info=ParseString(rt_temp,",")
    if S2I(rt_info[1])<>did_rts then do // something goofy
        if S2I(rt_info[1])=-19 then do // just got here early
        end
        else do
            ShowMessage("Unexpected CSG Summary Info.")
            RunDBox("G30 Show Array","CSG Summary",rt_info)
            err_lev=7
        end
    end
    rt_temp=ReadLine(lnk_file)
    // throw away the -19

    // Now write out the LP matrix files
    // =================================

    WriteLine(lp1_fh,String(read_lks))
    for aa=1 to read_lks do
        WriteLine(lp1_fh,lp_arr[aa]+" "+String(Sqrt(imp_arr[aa])))
    end

    WriteLine(lp2_fh,Format(obsno,"0000")+","+String(read_lks))

    end // big route loop -- no indent
    // if route header is usable and not -39

    UpdateProgressBar("Read "+String(did_obs)+" of "+String(tot_obs)
        +" observations.",Floor(100*did_obs/tot_obs))
end // giant loop over observations

DestroyProgressBar()
DisableProgressBar()

timenow=GetDateAndTime()
WriteLine(lp2_fh,"")
WriteLine(lp2_fh,"Run finished at "+timenow)
WriteLine(lp2_fh,"")

CloseFile(lp1_fh)
CloseFile(lp2_fh)
CloseFile(lnk_file)

ShowMessage("Macro finished with error level "+String(err_lev) )
```

```
endMacro
```

D
Estimation Code

This appendix contains the code to perform estimation of the PSL and CNL route choice models. The code presented in section D.1 was used to estimate Model 4 of Table 5-5. The code presented in section D.2 was used to estimate the model of Table 5-9 with μ_{m} defined by Equation 5-3.

Also, sample code to apply CNL models to the scenarios described in section 5-6 is shown below in section D.3. This particular example was used to apply the first CNL model of Table 5-9 (no Path Size, μ_{m} defined by Equation 5-1) to the New Mystic River Bridge Scenario.

D.1 Sample ALOGIT Program for PSL

Please note that the reader is referred to Hague Consulting Group (1995) for interpretation of the ALOGIT commands in each section of the following file.

```
Model F4r fu: dst, fft; ln(del)*inc; 3 tolls; nmrt; 2 lbls; IAP:NK and %nmrt
---------------------
- control lines
- all alts available; use all records

stats -1 2 0
nonav 3,803 4,804
nonav 5,805 6,806 7,807 8,808 9,809 10,810 11,811 12,812 13,813 14,814
nonav 15,815 16,816 17,817 18,818 19,819 20,820 21,821 22,822 23,823
nonav 24,824 25,825 26,826 27,827 28,828 29,829 30,830 31,831 32,832
nonav 33,833 34,834 35,835 36,836 37,837 38,838 39,839 40,840 41,841
nonav 42,842 43,843 44,844 45,845 46,846 47,847 48,848 49,849 50,850
nonav 51,851
l-s-m 499
stop 50

end
```

```
-------------------
- coefficients

1 dist
6 fft

16 masspike
17 tobin
18 sumner

171 im-lndel
172 i24lndel
175 i56lndel

60 nr_time

80 l_ldist
82 l_lct

404 ln_a_pn

499 pathsize f 1.0
-------------------
- transformations

id=d6
choice=d7

exclude=ifeq(d6,1043)
- less than 80 percent overlap

- use ps_def = 1 for distance, 2 for fft
ps_def = 2

sil=0.000001
- prevents underflow and logs of zero

lin_nk=(d17-0.0012)*2.572
- scaled network knowledge

- Income dummies
- there are six categories from the survey
- (plus -1 = missing = no response)
- 1 = less than $25,000 per year (household; before tax)
- 2 = $25,000 to $49,999
- 3 = $50,000 to $74,999
- 4 = $75,000 to $99,999
- 5 = $100,000 to $149,999
- 6 = $150,000 or more

- inc1 = ifeq(d9,1)
- no one in income category 1
inc2 = ifeq(d9,2)
inc3 = ifeq(d9,3)
inc4 = ifeq(d9,4)
inc5 = ifeq(d9,5)
inc6 = ifeq(d9,6)
```

```
inc24 = inc2+inc3+inc4
- inc56 = inc5+inc6
- constrained coefficients shown below in utilities

inc_miss=1-inc24-inc5-inc6

- availability dummies
- alt k unavailable if data(800+k) = 0
- alts 1 and 2 always available
d803 = ifge( d8, 3 )
d804 = ifge( d8, 4 )
d805 = ifge( d8, 5 )
d806 = ifge( d8, 6 )
d807 = ifge( d8, 7 )
d808 = ifge( d8, 8 )
d809 = ifge( d8, 9 )
d810 = ifge( d8, 10 )
d811 = ifge( d8, 11 )
d812 = ifge( d8, 12 )
d813 = ifge( d8, 13 )
d814 = ifge( d8, 14 )
d815 = ifge( d8, 15 )
d816 = ifge( d8, 16 )
d817 = ifge( d8, 17 )
d818 = ifge( d8, 18 )
d819 = ifge( d8, 19 )
d820 = ifge( d8, 20 )
d821 = ifge( d8, 21 )
d822 = ifge( d8, 22 )
d823 = ifge( d8, 23 )
d824 = ifge( d8, 24 )
d825 = ifge( d8, 25 )
d826 = ifge( d8, 26 )
d827 = ifge( d8, 27 )
d828 = ifge( d8, 28 )
d829 = ifge( d8, 29 )
d830 = ifge( d8, 30 )
d831 = ifge( d8, 31 )
d832 = ifge( d8, 32 )
d833 = ifge( d8, 33 )
d834 = ifge( d8, 34 )
d835 = ifge( d8, 35 )
d836 = ifge( d8, 36 )
d837 = ifge( d8, 37 )
d838 = ifge( d8, 38 )
d839 = ifge( d8, 39 )
d840 = ifge( d8, 40 )
d841 = ifge( d8, 41 )
d842 = ifge( d8, 42 )
d843 = ifge( d8, 43 )
d844 = ifge( d8, 44 )
d845 = ifge( d8, 45 )
d846 = ifge( d8, 46 )
d847 = ifge( d8, 47 )
d848 = ifge( d8, 48 )
d849 = ifge( d8, 49 )
```

```
d850 = ifge( d8, 50 )
d851 = ifge( d8, 51 )

- ins_orig = ifgt( ifeq(d2, 136, 531, 656, 700) , 0 )

- North Shore dummy

NShore = ifgt( ifeq(d2, 260,281,282,297,303,363,364,367,389,393,397,
                        400,419,422,435,436,501,504,512,515,612,615,
                        625,630,656,661,792,806) , 0 )

- Tobin Bridge Dummy defs

Tobin1 = NShore AND ifeq( d50 , 0.50 )
Tobin2 = NShore AND ifeq( d65 , 0.50 )
Tobin3 = NShore AND ifeq( d80 , 0.50 )
Tobin4 = NShore AND ifeq( d95 , 0.50 )
Tobin5 = NShore AND ifeq( d110 , 0.50 )
Tobin6 = NShore AND ifeq( d125 , 0.50 )
Tobin7 = NShore AND ifeq( d140 , 0.50 )
Tobin8 = NShore AND ifeq( d155 , 0.50 )
Tobin9 = NShore AND ifeq( d170 , 0.50 )
Tobin10 = NShore AND ifeq( d185 , 0.50 )
Tobin11 = NShore AND ifeq( d200 , 0.50 )
Tobin12 = NShore AND ifeq( d215 , 0.50 )
Tobin13 = NShore AND ifeq( d230 , 0.50 )
Tobin14 = NShore AND ifeq( d245 , 0.50 )
Tobin15 = NShore AND ifeq( d260 , 0.50 )
Tobin16 = NShore AND ifeq( d275 , 0.50 )
Tobin17 = NShore AND ifeq( d290 , 0.50 )
Tobin18 = NShore AND ifeq( d305 , 0.50 )
Tobin19 = NShore AND ifeq( d320 , 0.50 )
Tobin20 = NShore AND ifeq( d335 , 0.50 )
Tobin21 = NShore AND ifeq( d350 , 0.50 )
Tobin22 = NShore AND ifeq( d365 , 0.50 )
Tobin23 = NShore AND ifeq( d380 , 0.50 )
Tobin24 = NShore AND ifeq( d395 , 0.50 )
Tobin25 = NShore AND ifeq( d410 , 0.50 )
Tobin26 = NShore AND ifeq( d425 , 0.50 )
Tobin27 = NShore AND ifeq( d440 , 0.50 )
Tobin28 = NShore AND ifeq( d455 , 0.50 )
Tobin29 = NShore AND ifeq( d470 , 0.50 )
Tobin30 = NShore AND ifeq( d485 , 0.50 )
Tobin31 = NShore AND ifeq( d500 , 0.50 )
Tobin32 = NShore AND ifeq( d515 , 0.50 )
Tobin33 = NShore AND ifeq( d530 , 0.50 )
Tobin34 = NShore AND ifeq( d545 , 0.50 )
Tobin35 = NShore AND ifeq( d560 , 0.50 )
Tobin36 = NShore AND ifeq( d575 , 0.50 )
Tobin37 = NShore AND ifeq( d590 , 0.50 )
Tobin38 = NShore AND ifeq( d605 , 0.50 )
Tobin39 = NShore AND ifeq( d620 , 0.50 )
Tobin40 = NShore AND ifeq( d635 , 0.50 )
Tobin41 = NShore AND ifeq( d650 , 0.50 )
Tobin42 = NShore AND ifeq( d665 , 0.50 )
Tobin43 = NShore AND ifeq( d680 , 0.50 )
Tobin44 = NShore AND ifeq( d695 , 0.50 )
```

```
Tobin45 = NShore AND ifeq( d710 , 0.50 )
Tobin46 = NShore AND ifeq( d725 , 0.50 )
Tobin47 = NShore AND ifeq( d740 , 0.50 )
Tobin48 = NShore AND ifeq( d755 , 0.50 )
Tobin49 = NShore AND ifeq( d770 , 0.50 )
Tobin50 = NShore AND ifeq( d785 , 0.50 )
Tobin51 = NShore AND ifeq( d800 , 0.50 )

- Sumner Tunnel Dummy defs

Sumner1 = NShore AND ifeq( d50 , 1.00 )
Sumner2 = NShore AND ifeq( d65 , 1.00 )
Sumner3 = NShore AND ifeq( d80 , 1.00 )
Sumner4 = NShore AND ifeq( d95 , 1.00 )
Sumner5 = NShore AND ifeq( d110 , 1.00 )
Sumner6 = NShore AND ifeq( d125 , 1.00 )
Sumner7 = NShore AND ifeq( d140 , 1.00 )
Sumner8 = NShore AND ifeq( d155 , 1.00 )
Sumner9 = NShore AND ifeq( d170 , 1.00 )
Sumner10 = NShore AND ifeq( d185 , 1.00 )
Sumner11 = NShore AND ifeq( d200 , 1.00 )
Sumner12 = NShore AND ifeq( d215 , 1.00 )
Sumner13 = NShore AND ifeq( d230 , 1.00 )
Sumner14 = NShore AND ifeq( d245 , 1.00 )
Sumner15 = NShore AND ifeq( d260 , 1.00 )
Sumner16 = NShore AND ifeq( d275 , 1.00 )
Sumner17 = NShore AND ifeq( d290 , 1.00 )
Sumner18 = NShore AND ifeq( d305 , 1.00 )
Sumner19 = NShore AND ifeq( d320 , 1.00 )
Sumner20 = NShore AND ifeq( d335 , 1.00 )
Sumner21 = NShore AND ifeq( d350 , 1.00 )
Sumner22 = NShore AND ifeq( d365 , 1.00 )
Sumner23 = NShore AND ifeq( d380 , 1.00 )
Sumner24 = NShore AND ifeq( d395 , 1.00 )
Sumner25 = NShore AND ifeq( d410 , 1.00 )
Sumner26 = NShore AND ifeq( d425 , 1.00 )
Sumner27 = NShore AND ifeq( d440 , 1.00 )
Sumner28 = NShore AND ifeq( d455 , 1.00 )
Sumner29 = NShore AND ifeq( d470 , 1.00 )
Sumner30 = NShore AND ifeq( d485 , 1.00 )
Sumner31 = NShore AND ifeq( d500 , 1.00 )
Sumner32 = NShore AND ifeq( d515 , 1.00 )
Sumner33 = NShore AND ifeq( d530 , 1.00 )
Sumner34 = NShore AND ifeq( d545 , 1.00 )
Sumner35 = NShore AND ifeq( d560 , 1.00 )
Sumner36 = NShore AND ifeq( d575 , 1.00 )
Sumner37 = NShore AND ifeq( d590 , 1.00 )
Sumner38 = NShore AND ifeq( d605 , 1.00 )
Sumner39 = NShore AND ifeq( d620 , 1.00 )
Sumner40 = NShore AND ifeq( d635 , 1.00 )
Sumner41 = NShore AND ifeq( d650 , 1.00 )
Sumner42 = NShore AND ifeq( d665 , 1.00 )
Sumner43 = NShore AND ifeq( d680 , 1.00 )
Sumner44 = NShore AND ifeq( d695 , 1.00 )
Sumner45 = NShore AND ifeq( d710 , 1.00 )
Sumner46 = NShore AND ifeq( d725 , 1.00 )
Sumner47 = NShore AND ifeq( d740 , 1.00 )
```

```
Sumner48 = NShore AND ifeq( d755 , 1.00 )
Sumner49 = NShore AND ifeq( d770 , 1.00 )
Sumner50 = NShore AND ifeq( d785 , 1.00 )
Sumner51 = NShore AND ifeq( d800 , 1.00 )

- Mass Pike Dummy defs

MAPike1 = d50 AND ifeq( Tobin1, 0 ) AND ifeq( Sumner1, 0 )
MAPike2 = d65 AND ifeq( Tobin2, 0 ) AND ifeq( Sumner2, 0 )
MAPike3 = d80 AND ifeq( Tobin3, 0 ) AND ifeq( Sumner3, 0 )
MAPike4 = d95 AND ifeq( Tobin4, 0 ) AND ifeq( Sumner4, 0 )
MAPike5 = d110 AND ifeq( Tobin5, 0 ) AND ifeq( Sumner5, 0 )
MAPike6 = d125 AND ifeq( Tobin6, 0 ) AND ifeq( Sumner6, 0 )
MAPike7 = d140 AND ifeq( Tobin7, 0 ) AND ifeq( Sumner7, 0 )
MAPike8 = d155 AND ifeq( Tobin8, 0 ) AND ifeq( Sumner8, 0 )
MAPike9 = d170 AND ifeq( Tobin9, 0 ) AND ifeq( Sumner9, 0 )
MAPike10 = d185 AND ifeq( Tobin10, 0 ) AND ifeq( Sumner10, 0 )
MAPike11 = d200 AND ifeq( Tobin11, 0 ) AND ifeq( Sumner11, 0 )
MAPike12 = d215 AND ifeq( Tobin12, 0 ) AND ifeq( Sumner12, 0 )
MAPike13 = d230 AND ifeq( Tobin13, 0 ) AND ifeq( Sumner13, 0 )
MAPike14 = d245 AND ifeq( Tobin14, 0 ) AND ifeq( Sumner14, 0 )
MAPike15 = d260 AND ifeq( Tobin15, 0 ) AND ifeq( Sumner15, 0 )
MAPike16 = d275 AND ifeq( Tobin16, 0 ) AND ifeq( Sumner16, 0 )
MAPike17 = d290 AND ifeq( Tobin17, 0 ) AND ifeq( Sumner17, 0 )
MAPike18 = d305 AND ifeq( Tobin18, 0 ) AND ifeq( Sumner18, 0 )
MAPike19 = d320 AND ifeq( Tobin19, 0 ) AND ifeq( Sumner19, 0 )
MAPike20 = d335 AND ifeq( Tobin20, 0 ) AND ifeq( Sumner20, 0 )
MAPike21 = d350 AND ifeq( Tobin21, 0 ) AND ifeq( Sumner21, 0 )
MAPike22 = d365 AND ifeq( Tobin22, 0 ) AND ifeq( Sumner22, 0 )
MAPike23 = d380 AND ifeq( Tobin23, 0 ) AND ifeq( Sumner23, 0 )
MAPike24 = d395 AND ifeq( Tobin24, 0 ) AND ifeq( Sumner24, 0 )
MAPike25 = d410 AND ifeq( Tobin25, 0 ) AND ifeq( Sumner25, 0 )
MAPike26 = d425 AND ifeq( Tobin26, 0 ) AND ifeq( Sumner26, 0 )
MAPike27 = d440 AND ifeq( Tobin27, 0 ) AND ifeq( Sumner27, 0 )
MAPike28 = d455 AND ifeq( Tobin28, 0 ) AND ifeq( Sumner28, 0 )
MAPike29 = d470 AND ifeq( Tobin29, 0 ) AND ifeq( Sumner29, 0 )
MAPike30 = d485 AND ifeq( Tobin30, 0 ) AND ifeq( Sumner30, 0 )
MAPike31 = d500 AND ifeq( Tobin31, 0 ) AND ifeq( Sumner31, 0 )
MAPike32 = d515 AND ifeq( Tobin32, 0 ) AND ifeq( Sumner32, 0 )
MAPike33 = d530 AND ifeq( Tobin33, 0 ) AND ifeq( Sumner33, 0 )
MAPike34 = d545 AND ifeq( Tobin34, 0 ) AND ifeq( Sumner34, 0 )
MAPike35 = d560 AND ifeq( Tobin35, 0 ) AND ifeq( Sumner35, 0 )
MAPike36 = d575 AND ifeq( Tobin36, 0 ) AND ifeq( Sumner36, 0 )
MAPike37 = d590 AND ifeq( Tobin37, 0 ) AND ifeq( Sumner37, 0 )
MAPike38 = d605 AND ifeq( Tobin38, 0 ) AND ifeq( Sumner38, 0 )
MAPike39 = d620 AND ifeq( Tobin39, 0 ) AND ifeq( Sumner39, 0 )
MAPike40 = d635 AND ifeq( Tobin40, 0 ) AND ifeq( Sumner40, 0 )
MAPike41 = d650 AND ifeq( Tobin41, 0 ) AND ifeq( Sumner41, 0 )
MAPike42 = d665 AND ifeq( Tobin42, 0 ) AND ifeq( Sumner42, 0 )
MAPike43 = d680 AND ifeq( Tobin43, 0 ) AND ifeq( Sumner43, 0 )
MAPike44 = d695 AND ifeq( Tobin44, 0 ) AND ifeq( Sumner44, 0 )
MAPike45 = d710 AND ifeq( Tobin45, 0 ) AND ifeq( Sumner45, 0 )
MAPike46 = d725 AND ifeq( Tobin46, 0 ) AND ifeq( Sumner46, 0 )
MAPike47 = d740 AND ifeq( Tobin47, 0 ) AND ifeq( Sumner47, 0 )
MAPike48 = d755 AND ifeq( Tobin48, 0 ) AND ifeq( Sumner48, 0 )
MAPike49 = d770 AND ifeq( Tobin49, 0 ) AND ifeq( Sumner49, 0 )
MAPike50 = d785 AND ifeq( Tobin50, 0 ) AND ifeq( Sumner50, 0 )
```

```
MAPike51 = d800 AND ifeq( Tobin51, 0 ) AND ifeq( Sumner51, 0 )

- delay calculations: ct - fft
delay1 = d40 - d39
delay2 = d55 - d54
delay3 = d70 - d69
delay4 = d85 - d84
delay5 = d100 - d99
delay6 = d115 - d114
delay7 = d130 - d129
delay8 = d145 - d144
delay9 = d160 - d159
delay10 = d175 - d174
delay11 = d190 - d189
delay12 = d205 - d204
delay13 = d220 - d219
delay14 = d235 - d234
delay15 = d250 - d249
delay16 = d265 - d264
delay17 = d280 - d279
delay18 = d295 - d294
delay19 = d310 - d309
delay20 = d325 - d324
delay21 = d340 - d339
delay22 = d355 - d354
delay23 = d370 - d369
delay24 = d385 - d384
delay25 = d400 - d399
delay26 = d415 - d414
delay27 = d430 - d429
delay28 = d445 - d444
delay29 = d460 - d459
delay30 = d475 - d474
delay31 = d490 - d489
delay32 = d505 - d504
delay33 = d520 - d519
delay34 = d535 - d534
delay35 = d550 - d549
delay36 = d565 - d564
delay37 = d580 - d579
delay38 = d595 - d594
delay39 = d610 - d609
delay40 = d625 - d624
delay41 = d640 - d639
delay42 = d655 - d654
delay43 = d670 - d669
delay44 = d685 - d684
delay45 = d700 - d699
delay46 = d715 - d714
delay47 = d730 - d729
delay48 = d745 - d744
delay49 = d760 - d759
delay50 = d775 - d774
delay51 = d790 - d789

- log of delay
lndel1 = log(max(sil,delay1))
```

```
lndel2 = log(max(sil,delay2))
lndel3 = log(max(sil,delay3))
lndel4 = log(max(sil,delay4))
lndel5 = log(max(sil,delay5))
lndel6 = log(max(sil,delay6))
lndel7 = log(max(sil,delay7))
lndel8 = log(max(sil,delay8))
lndel9 = log(max(sil,delay9))
lndel10 = log(max(sil,delay10))
lndel11 = log(max(sil,delay11))
lndel12 = log(max(sil,delay12))
lndel13 = log(max(sil,delay13))
lndel14 = log(max(sil,delay14))
lndel15 = log(max(sil,delay15))
lndel16 = log(max(sil,delay16))
lndel17 = log(max(sil,delay17))
lndel18 = log(max(sil,delay18))
lndel19 = log(max(sil,delay19))
lndel20 = log(max(sil,delay20))
lndel21 = log(max(sil,delay21))
lndel22 = log(max(sil,delay22))
lndel23 = log(max(sil,delay23))
lndel24 = log(max(sil,delay24))
lndel25 = log(max(sil,delay25))
lndel26 = log(max(sil,delay26))
lndel27 = log(max(sil,delay27))
lndel28 = log(max(sil,delay28))
lndel29 = log(max(sil,delay29))
lndel30 = log(max(sil,delay30))
lndel31 = log(max(sil,delay31))
lndel32 = log(max(sil,delay32))
lndel33 = log(max(sil,delay33))
lndel34 = log(max(sil,delay34))
lndel35 = log(max(sil,delay35))
lndel36 = log(max(sil,delay36))
lndel37 = log(max(sil,delay37))
lndel38 = log(max(sil,delay38))
lndel39 = log(max(sil,delay39))
lndel40 = log(max(sil,delay40))
lndel41 = log(max(sil,delay41))
lndel42 = log(max(sil,delay42))
lndel43 = log(max(sil,delay43))
lndel44 = log(max(sil,delay44))
lndel45 = log(max(sil,delay45))
lndel46 = log(max(sil,delay46))
lndel47 = log(max(sil,delay47))
lndel48 = log(max(sil,delay48))
lndel49 = log(max(sil,delay49))
lndel50 = log(max(sil,delay50))
lndel51 = log(max(sil,delay51))

- now, income and log of delay interactions
- income is missing
imldl1 = inc_miss*lndel1
imldl2 = inc_miss*lndel2
imldl3 = inc_miss*lndel3
imldl4 = inc_miss*lndel4
```

```
imldl5 = inc_miss*lndel5
imldl6 = inc_miss*lndel6
imldl7 = inc_miss*lndel7
imldl8 = inc_miss*lndel8
imldl9 = inc_miss*lndel9
imldl10 = inc_miss*lndel10
imldl11 = inc_miss*lndel11
imldl12 = inc_miss*lndel12
imldl13 = inc_miss*lndel13
imldl14 = inc_miss*lndel14
imldl15 = inc_miss*lndel15
imldl16 = inc_miss*lndel16
imldl17 = inc_miss*lndel17
imldl18 = inc_miss*lndel18
imldl19 = inc_miss*lndel19
imldl20 = inc_miss*lndel20
imldl21 = inc_miss*lndel21
imldl22 = inc_miss*lndel22
imldl23 = inc_miss*lndel23
imldl24 = inc_miss*lndel24
imldl25 = inc_miss*lndel25
imldl26 = inc_miss*lndel26
imldl27 = inc_miss*lndel27
imldl28 = inc_miss*lndel28
imldl29 = inc_miss*lndel29
imldl30 = inc_miss*lndel30
imldl31 = inc_miss*lndel31
imldl32 = inc_miss*lndel32
imldl33 = inc_miss*lndel33
imldl34 = inc_miss*lndel34
imldl35 = inc_miss*lndel35
imldl36 = inc_miss*lndel36
imldl37 = inc_miss*lndel37
imldl38 = inc_miss*lndel38
imldl39 = inc_miss*lndel39
imldl40 = inc_miss*lndel40
imldl41 = inc_miss*lndel41
imldl42 = inc_miss*lndel42
imldl43 = inc_miss*lndel43
imldl44 = inc_miss*lndel44
imldl45 = inc_miss*lndel45
imldl46 = inc_miss*lndel46
imldl47 = inc_miss*lndel47
imldl48 = inc_miss*lndel48
imldl49 = inc_miss*lndel49
imldl50 = inc_miss*lndel50
imldl51 = inc_miss*lndel51

- income groups 2-4
i2ldl1 = inc24*lndel1
i2ldl2 = inc24*lndel2
i2ldl3 = inc24*lndel3
i2ldl4 = inc24*lndel4
i2ldl5 = inc24*lndel5
i2ldl6 = inc24*lndel6
i2ldl7 = inc24*lndel7
i2ldl8 = inc24*lndel8
```

```
i2ldl9 = inc24*lndel9
i2ldl10 = inc24*lndel10
i2ldl11 = inc24*lndel11
i2ldl12 = inc24*lndel12
i2ldl13 = inc24*lndel13
i2ldl14 = inc24*lndel14
i2ldl15 = inc24*lndel15
i2ldl16 = inc24*lndel16
i2ldl17 = inc24*lndel17
i2ldl18 = inc24*lndel18
i2ldl19 = inc24*lndel19
i2ldl20 = inc24*lndel20
i2ldl21 = inc24*lndel21
i2ldl22 = inc24*lndel22
i2ldl23 = inc24*lndel23
i2ldl24 = inc24*lndel24
i2ldl25 = inc24*lndel25
i2ldl26 = inc24*lndel26
i2ldl27 = inc24*lndel27
i2ldl28 = inc24*lndel28
i2ldl29 = inc24*lndel29
i2ldl30 = inc24*lndel30
i2ldl31 = inc24*lndel31
i2ldl32 = inc24*lndel32
i2ldl33 = inc24*lndel33
i2ldl34 = inc24*lndel34
i2ldl35 = inc24*lndel35
i2ldl36 = inc24*lndel36
i2ldl37 = inc24*lndel37
i2ldl38 = inc24*lndel38
i2ldl39 = inc24*lndel39
i2ldl40 = inc24*lndel40
i2ldl41 = inc24*lndel41
i2ldl42 = inc24*lndel42
i2ldl43 = inc24*lndel43
i2ldl44 = inc24*lndel44
i2ldl45 = inc24*lndel45
i2ldl46 = inc24*lndel46
i2ldl47 = inc24*lndel47
i2ldl48 = inc24*lndel48
i2ldl49 = inc24*lndel49
i2ldl50 = inc24*lndel50
i2ldl51 = inc24*lndel51

- income group 5
i5ldl1 = inc5*lndel1
i5ldl2 = inc5*lndel2
i5ldl3 = inc5*lndel3
i5ldl4 = inc5*lndel4
i5ldl5 = inc5*lndel5
i5ldl6 = inc5*lndel6
i5ldl7 = inc5*lndel7
i5ldl8 = inc5*lndel8
i5ldl9 = inc5*lndel9
i5ldl10 = inc5*lndel10
i5ldl11 = inc5*lndel11
i5ldl12 = inc5*lndel12
```

```
i5ldl13 = inc5*lndel13
i5ldl14 = inc5*lndel14
i5ldl15 = inc5*lndel15
i5ldl16 = inc5*lndel16
i5ldl17 = inc5*lndel17
i5ldl18 = inc5*lndel18
i5ldl19 = inc5*lndel19
i5ldl20 = inc5*lndel20
i5ldl21 = inc5*lndel21
i5ldl22 = inc5*lndel22
i5ldl23 = inc5*lndel23
i5ldl24 = inc5*lndel24
i5ldl25 = inc5*lndel25
i5ldl26 = inc5*lndel26
i5ldl27 = inc5*lndel27
i5ldl28 = inc5*lndel28
i5ldl29 = inc5*lndel29
i5ldl30 = inc5*lndel30
i5ldl31 = inc5*lndel31
i5ldl32 = inc5*lndel32
i5ldl33 = inc5*lndel33
i5ldl34 = inc5*lndel34
i5ldl35 = inc5*lndel35
i5ldl36 = inc5*lndel36
i5ldl37 = inc5*lndel37
i5ldl38 = inc5*lndel38
i5ldl39 = inc5*lndel39
i5ldl40 = inc5*lndel40
i5ldl41 = inc5*lndel41
i5ldl42 = inc5*lndel42
i5ldl43 = inc5*lndel43
i5ldl44 = inc5*lndel44
i5ldl45 = inc5*lndel45
i5ldl46 = inc5*lndel46
i5ldl47 = inc5*lndel47
i5ldl48 = inc5*lndel48
i5ldl49 = inc5*lndel49
i5ldl50 = inc5*lndel50
i5ldl51 = inc5*lndel51

- income group 6
i6ldl1 = inc6*lndel1
i6ldl2 = inc6*lndel2
i6ldl3 = inc6*lndel3
i6ldl4 = inc6*lndel4
i6ldl5 = inc6*lndel5
i6ldl6 = inc6*lndel6
i6ldl7 = inc6*lndel7
i6ldl8 = inc6*lndel8
i6ldl9 = inc6*lndel9
i6ldl10 = inc6*lndel10
i6ldl11 = inc6*lndel11
i6ldl12 = inc6*lndel12
i6ldl13 = inc6*lndel13
i6ldl14 = inc6*lndel14
i6ldl15 = inc6*lndel15
i6ldl16 = inc6*lndel16
```

```
i6ldl17 = inc6*lndel17
i6ldl18 = inc6*lndel18
i6ldl19 = inc6*lndel19
i6ldl20 = inc6*lndel20
i6ldl21 = inc6*lndel21
i6ldl22 = inc6*lndel22
i6ldl23 = inc6*lndel23
i6ldl24 = inc6*lndel24
i6ldl25 = inc6*lndel25
i6ldl26 = inc6*lndel26
i6ldl27 = inc6*lndel27
i6ldl28 = inc6*lndel28
i6ldl29 = inc6*lndel29
i6ldl30 = inc6*lndel30
i6ldl31 = inc6*lndel31
i6ldl32 = inc6*lndel32
i6ldl33 = inc6*lndel33
i6ldl34 = inc6*lndel34
i6ldl35 = inc6*lndel35
i6ldl36 = inc6*lndel36
i6ldl37 = inc6*lndel37
i6ldl38 = inc6*lndel38
i6ldl39 = inc6*lndel39
i6ldl40 = inc6*lndel40
i6ldl41 = inc6*lndel41
i6ldl42 = inc6*lndel42
i6ldl43 = inc6*lndel43
i6ldl44 = inc6*lndel44
i6ldl45 = inc6*lndel45
i6ldl46 = inc6*lndel46
i6ldl47 = inc6*lndel47
i6ldl48 = inc6*lndel48
i6ldl49 = inc6*lndel49
i6ldl50 = inc6*lndel50
i6ldl51 = inc6*lndel51

- code distance for minimizing
dist1 = d38
dist2 = d53
dist3 = d68 + 299*(1-d803)
dist4 = d83 + 299*(1-d804)
dist5 = d98 + 299*(1-d805)
dist6 = d113 + 299*(1-d806)
dist7 = d128 + 299*(1-d807)
dist8 = d143 + 299*(1-d808)
dist9 = d158 + 299*(1-d809)
dist10 = d173 + 299*(1-d810)
dist11 = d188 + 299*(1-d811)
dist12 = d203 + 299*(1-d812)
dist13 = d218 + 299*(1-d813)
dist14 = d233 + 299*(1-d814)
dist15 = d248 + 299*(1-d815)
dist16 = d263 + 299*(1-d816)
dist17 = d278 + 299*(1-d817)
dist18 = d293 + 299*(1-d818)
dist19 = d308 + 299*(1-d819)
dist20 = d323 + 299*(1-d820)
```

```
dist21 = d338 + 299*(1-d821)
dist22 = d353 + 299*(1-d822)
dist23 = d368 + 299*(1-d823)
dist24 = d383 + 299*(1-d824)
dist25 = d398 + 299*(1-d825)
dist26 = d413 + 299*(1-d826)
dist27 = d428 + 299*(1-d827)
dist28 = d443 + 299*(1-d828)
dist29 = d458 + 299*(1-d829)
dist30 = d473 + 299*(1-d830)
dist31 = d488 + 299*(1-d831)
dist32 = d503 + 299*(1-d832)
dist33 = d518 + 299*(1-d833)
dist34 = d533 + 299*(1-d834)
dist35 = d548 + 299*(1-d835)
dist36 = d563 + 299*(1-d836)
dist37 = d578 + 299*(1-d837)
dist38 = d593 + 299*(1-d838)
dist39 = d608 + 299*(1-d839)
dist40 = d623 + 299*(1-d840)
dist41 = d638 + 299*(1-d841)
dist42 = d653 + 299*(1-d842)
dist43 = d668 + 299*(1-d843)
dist44 = d683 + 299*(1-d844)
dist45 = d698 + 299*(1-d845)
dist46 = d713 + 299*(1-d846)
dist47 = d728 + 299*(1-d847)
dist48 = d743 + 299*(1-d848)
dist49 = d758 + 299*(1-d849)
dist50 = d773 + 299*(1-d850)
dist51 = d788 + 299*(1-d851)

mindist=MIN( dist1, dist2, dist3, dist4, dist5, dist6, dist7, dist8,
             dist9, dist10, dist11, dist12, dist13, dist14, dist15,
             dist16, dist17, dist18, dist19, dist20, dist21, dist22,
             dist23, dist24, dist25, dist26, dist27, dist28, dist29,
             dist30, dist31, dist32, dist33, dist34, dist35, dist36,
             dist37, dist38, dist39, dist40, dist41, dist42, dist43,
             dist44, dist45, dist46, dist47, dist48, dist49, dist50,
             dist51 )
- luckily, ALOGIT accepts up to 51 arguments

LDL1     = ifeq( mindist, dist1  )
LDL2     = ifeq( mindist, dist2  )
LDL3     = ifeq( mindist, dist3  )
LDL4     = ifeq( mindist, dist4  )
LDL5     = ifeq( mindist, dist5  )
LDL6     = ifeq( mindist, dist6  )
LDL7     = ifeq( mindist, dist7  )
LDL8     = ifeq( mindist, dist8  )
LDL9     = ifeq( mindist, dist9  )
LDL10    = ifeq( mindist, dist10 )
LDL11    = ifeq( mindist, dist11 )
LDL12    = ifeq( mindist, dist12 )
LDL13    = ifeq( mindist, dist13 )
LDL14    = ifeq( mindist, dist14 )
LDL15    = ifeq( mindist, dist15 )
```

```
LDL16    = ifeq( mindist, dist16 )
LDL17    = ifeq( mindist, dist17 )
LDL18    = ifeq( mindist, dist18 )
LDL19    = ifeq( mindist, dist19 )
LDL20    = ifeq( mindist, dist20 )
LDL21    = ifeq( mindist, dist21 )
LDL22    = ifeq( mindist, dist22 )
LDL23    = ifeq( mindist, dist23 )
LDL24    = ifeq( mindist, dist24 )
LDL25    = ifeq( mindist, dist25 )
LDL26    = ifeq( mindist, dist26 )
LDL27    = ifeq( mindist, dist27 )
LDL28    = ifeq( mindist, dist28 )
LDL29    = ifeq( mindist, dist29 )
LDL30    = ifeq( mindist, dist30 )
LDL31    = ifeq( mindist, dist31 )
LDL32    = ifeq( mindist, dist32 )
LDL33    = ifeq( mindist, dist33 )
LDL34    = ifeq( mindist, dist34 )
LDL35    = ifeq( mindist, dist35 )
LDL36    = ifeq( mindist, dist36 )
LDL37    = ifeq( mindist, dist37 )
LDL38    = ifeq( mindist, dist38 )
LDL39    = ifeq( mindist, dist39 )
LDL40    = ifeq( mindist, dist40 )
LDL41    = ifeq( mindist, dist41 )
LDL42    = ifeq( mindist, dist42 )
LDL43    = ifeq( mindist, dist43 )
LDL44    = ifeq( mindist, dist44 )
LDL45    = ifeq( mindist, dist45 )
LDL46    = ifeq( mindist, dist46 )
LDL47    = ifeq( mindist, dist47 )
LDL48    = ifeq( mindist, dist48 )
LDL49    = ifeq( mindist, dist49 )
LDL50    = ifeq( mindist, dist50 )
LDL51    = ifeq( mindist, dist51 )

- recode free-flow time for utility
fft1 = d39
fft2 = d54
fft3 = d69 + 299*(1-d803)
fft4 = d84 + 299*(1-d804)
fft5 = d99 + 299*(1-d805)
fft6 = d114 + 299*(1-d806)
fft7 = d129 + 299*(1-d807)
fft8 = d144 + 299*(1-d808)
fft9 = d159 + 299*(1-d809)
fft10 = d174 + 299*(1-d810)
fft11 = d189 + 299*(1-d811)
fft12 = d204 + 299*(1-d812)
fft13 = d219 + 299*(1-d813)
fft14 = d234 + 299*(1-d814)
fft15 = d249 + 299*(1-d815)
fft16 = d264 + 299*(1-d816)
fft17 = d279 + 299*(1-d817)
fft18 = d294 + 299*(1-d818)
fft19 = d309 + 299*(1-d819)
```

```
fft20 = d324 + 299*(1-d820)
fft21 = d339 + 299*(1-d821)
fft22 = d354 + 299*(1-d822)
fft23 = d369 + 299*(1-d823)
fft24 = d384 + 299*(1-d824)
fft25 = d399 + 299*(1-d825)
fft26 = d414 + 299*(1-d826)
fft27 = d429 + 299*(1-d827)
fft28 = d444 + 299*(1-d828)
fft29 = d459 + 299*(1-d829)
fft30 = d474 + 299*(1-d830)
fft31 = d489 + 299*(1-d831)
fft32 = d504 + 299*(1-d832)
fft33 = d519 + 299*(1-d833)
fft34 = d534 + 299*(1-d834)
fft35 = d549 + 299*(1-d835)
fft36 = d564 + 299*(1-d836)
fft37 = d579 + 299*(1-d837)
fft38 = d594 + 299*(1-d838)
fft39 = d609 + 299*(1-d839)
fft40 = d624 + 299*(1-d840)
fft41 = d639 + 299*(1-d841)
fft42 = d654 + 299*(1-d842)
fft43 = d669 + 299*(1-d843)
fft44 = d684 + 299*(1-d844)
fft45 = d699 + 299*(1-d845)
fft46 = d714 + 299*(1-d846)
fft47 = d729 + 299*(1-d847)
fft48 = d744 + 299*(1-d848)
fft49 = d759 + 299*(1-d849)
fft50 = d774 + 299*(1-d850)
fft51 = d789 + 299*(1-d851)

- Least Congested Time Label dummy defs

- recode congested time for minimizing
ct1 = d40
ct2 = d55
ct3 = d70 + 9999*(1-d803)
ct4 = d85 + 9999*(1-d804)
ct5 = d100 + 9999*(1-d805)
ct6 = d115 + 9999*(1-d806)
ct7 = d130 + 9999*(1-d807)
ct8 = d145 + 9999*(1-d808)
ct9 = d160 + 9999*(1-d809)
ct10 = d175 + 9999*(1-d810)
ct11 = d190 + 9999*(1-d811)
ct12 = d205 + 9999*(1-d812)
ct13 = d220 + 9999*(1-d813)
ct14 = d235 + 9999*(1-d814)
ct15 = d250 + 9999*(1-d815)
ct16 = d265 + 9999*(1-d816)
ct17 = d280 + 9999*(1-d817)
ct18 = d295 + 9999*(1-d818)
ct19 = d310 + 9999*(1-d819)
ct20 = d325 + 9999*(1-d820)
ct21 = d340 + 9999*(1-d821)
```

```
ct22 = d355 + 9999*(1-d822)
ct23 = d370 + 9999*(1-d823)
ct24 = d385 + 9999*(1-d824)
ct25 = d400 + 9999*(1-d825)
ct26 = d415 + 9999*(1-d826)
ct27 = d430 + 9999*(1-d827)
ct28 = d445 + 9999*(1-d828)
ct29 = d460 + 9999*(1-d829)
ct30 = d475 + 9999*(1-d830)
ct31 = d490 + 9999*(1-d831)
ct32 = d505 + 9999*(1-d832)
ct33 = d520 + 9999*(1-d833)
ct34 = d535 + 9999*(1-d834)
ct35 = d550 + 9999*(1-d835)
ct36 = d565 + 9999*(1-d836)
ct37 = d580 + 9999*(1-d837)
ct38 = d595 + 9999*(1-d838)
ct39 = d610 + 9999*(1-d839)
ct40 = d625 + 9999*(1-d840)
ct41 = d640 + 9999*(1-d841)
ct42 = d655 + 9999*(1-d842)
ct43 = d670 + 9999*(1-d843)
ct44 = d685 + 9999*(1-d844)
ct45 = d700 + 9999*(1-d845)
ct46 = d715 + 9999*(1-d846)
ct47 = d730 + 9999*(1-d847)
ct48 = d745 + 9999*(1-d848)
ct49 = d760 + 9999*(1-d849)
ct50 = d775 + 9999*(1-d850)
ct51 = d790 + 9999*(1-d851)

minct=MIN( ct1, ct2, ct3, ct4, ct5, ct6, ct7, ct8, ct9, ct10, ct11,
           ct12, ct13, ct14, ct15, ct16, ct17, ct18, ct19, ct20, ct21,
           ct22, ct23, ct24, ct25, ct26, ct27, ct28, ct29, ct30, ct31,
           ct32, ct33, ct34, ct35, ct36, ct37, ct38, ct39, ct40, ct41,
           ct42, ct43, ct44, ct45, ct46, ct47, ct48, ct49, ct50, ct51 )

LCTL1    = ifeq( minct, ct1  )
LCTL2    = ifeq( minct, ct2  )
LCTL3    = ifeq( minct, ct3  )
LCTL4    = ifeq( minct, ct4  )
LCTL5    = ifeq( minct, ct5  )
LCTL6    = ifeq( minct, ct6  )
LCTL7    = ifeq( minct, ct7  )
LCTL8    = ifeq( minct, ct8  )
LCTL9    = ifeq( minct, ct9  )
LCTL10   = ifeq( minct, ct10 )
LCTL11   = ifeq( minct, ct11 )
LCTL12   = ifeq( minct, ct12 )
LCTL13   = ifeq( minct, ct13 )
LCTL14   = ifeq( minct, ct14 )
LCTL15   = ifeq( minct, ct15 )
LCTL16   = ifeq( minct, ct16 )
LCTL17   = ifeq( minct, ct17 )
LCTL18   = ifeq( minct, ct18 )
LCTL19   = ifeq( minct, ct19 )
LCTL20   = ifeq( minct, ct20 )
```

```
LCTL21   = ifeq( minct, ct21 )
LCTL22   = ifeq( minct, ct22 )
LCTL23   = ifeq( minct, ct23 )
LCTL24   = ifeq( minct, ct24 )
LCTL25   = ifeq( minct, ct25 )
LCTL26   = ifeq( minct, ct26 )
LCTL27   = ifeq( minct, ct27 )
LCTL28   = ifeq( minct, ct28 )
LCTL29   = ifeq( minct, ct29 )
LCTL30   = ifeq( minct, ct30 )
LCTL31   = ifeq( minct, ct31 )
LCTL32   = ifeq( minct, ct32 )
LCTL33   = ifeq( minct, ct33 )
LCTL34   = ifeq( minct, ct34 )
LCTL35   = ifeq( minct, ct35 )
LCTL36   = ifeq( minct, ct36 )
LCTL37   = ifeq( minct, ct37 )
LCTL38   = ifeq( minct, ct38 )
LCTL39   = ifeq( minct, ct39 )
LCTL40   = ifeq( minct, ct40 )
LCTL41   = ifeq( minct, ct41 )
LCTL42   = ifeq( minct, ct42 )
LCTL43   = ifeq( minct, ct43 )
LCTL44   = ifeq( minct, ct44 )
LCTL45   = ifeq( minct, ct45 )
LCTL46   = ifeq( minct, ct46 )
LCTL47   = ifeq( minct, ct47 )
LCTL48   = ifeq( minct, ct48 )
LCTL49   = ifeq( minct, ct49 )
LCTL50   = ifeq( minct, ct50 )
LCTL51   = ifeq( minct, ct51 )

- calc percent numbered route time for maximizing
pnmr1 = d49 / d40
pnmr2 = d64 / d55
pnmr3 = recode( d803+1: 0, d79/d70 )
pnmr4 = recode( d804+1: 0, d94/d85 )
pnmr5 = recode( d805+1: 0, d109/d100 )
pnmr6 = recode( d806+1: 0, d124/d115 )
pnmr7 = recode( d807+1: 0, d139/d130 )
pnmr8 = recode( d808+1: 0, d154/d145 )
pnmr9 = recode( d809+1: 0, d169/d160 )
pnmr10 = recode( d810+1: 0, d184/d175 )
pnmr11 = recode( d811+1: 0, d199/d190 )
pnmr12 = recode( d812+1: 0, d214/d205 )
pnmr13 = recode( d813+1: 0, d229/d220 )
pnmr14 = recode( d814+1: 0, d244/d235 )
pnmr15 = recode( d815+1: 0, d259/d250 )
pnmr16 = recode( d816+1: 0, d274/d265 )
pnmr17 = recode( d817+1: 0, d289/d280 )
pnmr18 = recode( d818+1: 0, d304/d295 )
pnmr19 = recode( d819+1: 0, d319/d310 )
pnmr20 = recode( d820+1: 0, d334/d325 )
pnmr21 = recode( d821+1: 0, d349/d340 )
pnmr22 = recode( d822+1: 0, d364/d355 )
pnmr23 = recode( d823+1: 0, d379/d370 )
pnmr24 = recode( d824+1: 0, d394/d385 )
```

```
pnmr25 = recode( d825+1: 0, d409/d400 )
pnmr26 = recode( d826+1: 0, d424/d415 )
pnmr27 = recode( d827+1: 0, d439/d430 )
pnmr28 = recode( d828+1: 0, d454/d445 )
pnmr29 = recode( d829+1: 0, d469/d460 )
pnmr30 = recode( d830+1: 0, d484/d475 )
pnmr31 = recode( d831+1: 0, d499/d490 )
pnmr32 = recode( d832+1: 0, d514/d505 )
pnmr33 = recode( d833+1: 0, d529/d520 )
pnmr34 = recode( d834+1: 0, d544/d535 )
pnmr35 = recode( d835+1: 0, d559/d550 )
pnmr36 = recode( d836+1: 0, d574/d565 )
pnmr37 = recode( d837+1: 0, d589/d580 )
pnmr38 = recode( d838+1: 0, d604/d595 )
pnmr39 = recode( d839+1: 0, d619/d610 )
pnmr40 = recode( d840+1: 0, d634/d625 )
pnmr41 = recode( d841+1: 0, d649/d640 )
pnmr42 = recode( d842+1: 0, d664/d655 )
pnmr43 = recode( d843+1: 0, d679/d670 )
pnmr44 = recode( d844+1: 0, d694/d685 )
pnmr45 = recode( d845+1: 0, d709/d700 )
pnmr46 = recode( d846+1: 0, d724/d715 )
pnmr47 = recode( d847+1: 0, d739/d730 )
pnmr48 = recode( d848+1: 0, d754/d745 )
pnmr49 = recode( d849+1: 0, d769/d760 )
pnmr50 = recode( d850+1: 0, d784/d775 )
pnmr51 = recode( d851+1: 0, d799/d790 )

- IAP based on max pct num rt time
NRpnrt1 = 1 - pnmr1
NRpnrt2 = 1 - pnmr2
NRpnrt3 = 1 - pnmr3
NRpnrt4 = 1 - pnmr4
NRpnrt5 = 1 - pnmr5
NRpnrt6 = 1 - pnmr6
NRpnrt7 = 1 - pnmr7
NRpnrt8 = 1 - pnmr8
NRpnrt9 = 1 - pnmr9
NRpnrt10 = 1 - pnmr10
NRpnrt11 = 1 - pnmr11
NRpnrt12 = 1 - pnmr12
NRpnrt13 = 1 - pnmr13
NRpnrt14 = 1 - pnmr14
NRpnrt15 = 1 - pnmr15
NRpnrt16 = 1 - pnmr16
NRpnrt17 = 1 - pnmr17
NRpnrt18 = 1 - pnmr18
NRpnrt19 = 1 - pnmr19
NRpnrt20 = 1 - pnmr20
NRpnrt21 = 1 - pnmr21
NRpnrt22 = 1 - pnmr22
NRpnrt23 = 1 - pnmr23
NRpnrt24 = 1 - pnmr24
NRpnrt25 = 1 - pnmr25
NRpnrt26 = 1 - pnmr26
NRpnrt27 = 1 - pnmr27
NRpnrt28 = 1 - pnmr28
```

```
NRpnrt29 = 1 - pnmr29
NRpnrt30 = 1 - pnmr30
NRpnrt31 = 1 - pnmr31
NRpnrt32 = 1 - pnmr32
NRpnrt33 = 1 - pnmr33
NRpnrt34 = 1 - pnmr34
NRpnrt35 = 1 - pnmr35
NRpnrt36 = 1 - pnmr36
NRpnrt37 = 1 - pnmr37
NRpnrt38 = 1 - pnmr38
NRpnrt39 = 1 - pnmr39
NRpnrt40 = 1 - pnmr40
NRpnrt41 = 1 - pnmr41
NRpnrt42 = 1 - pnmr42
NRpnrt43 = 1 - pnmr43
NRpnrt44 = 1 - pnmr44
NRpnrt45 = 1 - pnmr45
NRpnrt46 = 1 - pnmr46
NRpnrt47 = 1 - pnmr47
NRpnrt48 = 1 - pnmr48
NRpnrt49 = 1 - pnmr49
NRpnrt50 = 1 - pnmr50
NRpnrt51 = 1 - pnmr51

lnapn1 = log( max(sil,1-(1-lin_nk)*NRpnrt1 ) )
lnapn2 = log( max(sil,1-(1-lin_nk)*NRpnrt2 ) )
lnapn3 = log( max(sil,1-(1-lin_nk)*NRpnrt3 ) )
lnapn4 = log( max(sil,1-(1-lin_nk)*NRpnrt4 ) )
lnapn5 = log( max(sil,1-(1-lin_nk)*NRpnrt5 ) )
lnapn6 = log( max(sil,1-(1-lin_nk)*NRpnrt6 ) )
lnapn7 = log( max(sil,1-(1-lin_nk)*NRpnrt7 ) )
lnapn8 = log( max(sil,1-(1-lin_nk)*NRpnrt8 ) )
lnapn9 = log( max(sil,1-(1-lin_nk)*NRpnrt9 ) )
lnapn10 = log( max(sil,1-(1-lin_nk)*NRpnrt10 ) )
lnapn11 = log( max(sil,1-(1-lin_nk)*NRpnrt11 ) )
lnapn12 = log( max(sil,1-(1-lin_nk)*NRpnrt12 ) )
lnapn13 = log( max(sil,1-(1-lin_nk)*NRpnrt13 ) )
lnapn14 = log( max(sil,1-(1-lin_nk)*NRpnrt14 ) )
lnapn15 = log( max(sil,1-(1-lin_nk)*NRpnrt15 ) )
lnapn16 = log( max(sil,1-(1-lin_nk)*NRpnrt16 ) )
lnapn17 = log( max(sil,1-(1-lin_nk)*NRpnrt17 ) )
lnapn18 = log( max(sil,1-(1-lin_nk)*NRpnrt18 ) )
lnapn19 = log( max(sil,1-(1-lin_nk)*NRpnrt19 ) )
lnapn20 = log( max(sil,1-(1-lin_nk)*NRpnrt20 ) )
lnapn21 = log( max(sil,1-(1-lin_nk)*NRpnrt21 ) )
lnapn22 = log( max(sil,1-(1-lin_nk)*NRpnrt22 ) )
lnapn23 = log( max(sil,1-(1-lin_nk)*NRpnrt23 ) )
lnapn24 = log( max(sil,1-(1-lin_nk)*NRpnrt24 ) )
lnapn25 = log( max(sil,1-(1-lin_nk)*NRpnrt25 ) )
lnapn26 = log( max(sil,1-(1-lin_nk)*NRpnrt26 ) )
lnapn27 = log( max(sil,1-(1-lin_nk)*NRpnrt27 ) )
lnapn28 = log( max(sil,1-(1-lin_nk)*NRpnrt28 ) )
lnapn29 = log( max(sil,1-(1-lin_nk)*NRpnrt29 ) )
lnapn30 = log( max(sil,1-(1-lin_nk)*NRpnrt30 ) )
lnapn31 = log( max(sil,1-(1-lin_nk)*NRpnrt31 ) )
lnapn32 = log( max(sil,1-(1-lin_nk)*NRpnrt32 ) )
lnapn33 = log( max(sil,1-(1-lin_nk)*NRpnrt33 ) )
```

```
lnapn34 = log( max(sil,1-(1-lin_nk)*NRpnrt34 ) )
lnapn35 = log( max(sil,1-(1-lin_nk)*NRpnrt35 ) )
lnapn36 = log( max(sil,1-(1-lin_nk)*NRpnrt36 ) )
lnapn37 = log( max(sil,1-(1-lin_nk)*NRpnrt37 ) )
lnapn38 = log( max(sil,1-(1-lin_nk)*NRpnrt38 ) )
lnapn39 = log( max(sil,1-(1-lin_nk)*NRpnrt39 ) )
lnapn40 = log( max(sil,1-(1-lin_nk)*NRpnrt40 ) )
lnapn41 = log( max(sil,1-(1-lin_nk)*NRpnrt41 ) )
lnapn42 = log( max(sil,1-(1-lin_nk)*NRpnrt42 ) )
lnapn43 = log( max(sil,1-(1-lin_nk)*NRpnrt43 ) )
lnapn44 = log( max(sil,1-(1-lin_nk)*NRpnrt44 ) )
lnapn45 = log( max(sil,1-(1-lin_nk)*NRpnrt45 ) )
lnapn46 = log( max(sil,1-(1-lin_nk)*NRpnrt46 ) )
lnapn47 = log( max(sil,1-(1-lin_nk)*NRpnrt47 ) )
lnapn48 = log( max(sil,1-(1-lin_nk)*NRpnrt48 ) )
lnapn49 = log( max(sil,1-(1-lin_nk)*NRpnrt49 ) )
lnapn50 = log( max(sil,1-(1-lin_nk)*NRpnrt50 ) )
lnapn51 = log( max(sil,1-(1-lin_nk)*NRpnrt51 ) )

- easier path size
- ps_def = 1 for distance, 2 for fft
pthsiz1 = recode( ps_def: d51 , d52 )
pthsiz2 = recode( ps_def: d66 , d67 )
pthsiz3 = recode( ps_def: d81 , d82 )
pthsiz4 = recode( ps_def: d96 , d97 )
pthsiz5 = recode( ps_def: d111 , d112 )
pthsiz6 = recode( ps_def: d126 , d127 )
pthsiz7 = recode( ps_def: d141 , d142 )
pthsiz8 = recode( ps_def: d156 , d157 )
pthsiz9 = recode( ps_def: d171 , d172 )
pthsiz10 = recode( ps_def: d186 , d187 )
pthsiz11 = recode( ps_def: d201 , d202 )
pthsiz12 = recode( ps_def: d216 , d217 )
pthsiz13 = recode( ps_def: d231 , d232 )
pthsiz14 = recode( ps_def: d246 , d247 )
pthsiz15 = recode( ps_def: d261 , d262 )
pthsiz16 = recode( ps_def: d276 , d277 )
pthsiz17 = recode( ps_def: d291 , d292 )
pthsiz18 = recode( ps_def: d306 , d307 )
pthsiz19 = recode( ps_def: d321 , d322 )
pthsiz20 = recode( ps_def: d336 , d337 )
pthsiz21 = recode( ps_def: d351 , d352 )
pthsiz22 = recode( ps_def: d366 , d367 )
pthsiz23 = recode( ps_def: d381 , d382 )
pthsiz24 = recode( ps_def: d396 , d397 )
pthsiz25 = recode( ps_def: d411 , d412 )
pthsiz26 = recode( ps_def: d426 , d427 )
pthsiz27 = recode( ps_def: d441 , d442 )
pthsiz28 = recode( ps_def: d456 , d457 )
pthsiz29 = recode( ps_def: d471 , d472 )
pthsiz30 = recode( ps_def: d486 , d487 )
pthsiz31 = recode( ps_def: d501 , d502 )
pthsiz32 = recode( ps_def: d516 , d517 )
pthsiz33 = recode( ps_def: d531 , d532 )
pthsiz34 = recode( ps_def: d546 , d547 )
pthsiz35 = recode( ps_def: d561 , d562 )
pthsiz36 = recode( ps_def: d576 , d577 )
```

```
pthsiz37 = recode( ps_def: d591 , d592 )
pthsiz38 = recode( ps_def: d606 , d607 )
pthsiz39 = recode( ps_def: d621 , d622 )
pthsiz40 = recode( ps_def: d636 , d637 )
pthsiz41 = recode( ps_def: d651 , d652 )
pthsiz42 = recode( ps_def: d666 , d667 )
pthsiz43 = recode( ps_def: d681 , d682 )
pthsiz44 = recode( ps_def: d696 , d697 )
pthsiz45 = recode( ps_def: d711 , d712 )
pthsiz46 = recode( ps_def: d726 , d727 )
pthsiz47 = recode( ps_def: d741 , d742 )
pthsiz48 = recode( ps_def: d756 , d757 )
pthsiz49 = recode( ps_def: d771 , d772 )
pthsiz50 = recode( ps_def: d786 , d787 )
pthsiz51 = recode( ps_def: d801 , d802 )

--------------------------------------
- utilities for model F4r
util1 = p1*dist1 + p6*fft1
      + p16*MAPike1 + p17*Tobin1 + p18*Sumner1
      + p171*imldl1 + p172*i2ldl1 + p175*i5ldl1 + p175*i6ldl1
      + p60*d49 + p80*LDL1 + p82*LCTL1
      + p404*lnapn1
util2 = p1*dist2 + p6*fft2
      + p16*MAPike2 + p17*Tobin2 + p18*Sumner2
      + p171*imldl2 + p172*i2ldl2 + p175*i5ldl2 + p175*i6ldl2
      + p60*d64 + p80*LDL2 + p82*LCTL2
      + p404*lnapn2
util3 = p1*dist3 + p6*fft3
      + p16*MAPike3 + p17*Tobin3 + p18*Sumner3
      + p171*imldl3 + p172*i2ldl3 + p175*i5ldl3 + p175*i6ldl3
      + p60*d79 + p80*LDL3 + p82*LCTL3
      + p404*lnapn3
util4 = p1*dist4 + p6*fft4
      + p16*MAPike4 + p17*Tobin4 + p18*Sumner4
      + p171*imldl4 + p172*i2ldl4 + p175*i5ldl4 + p175*i6ldl4
      + p60*d94 + p80*LDL4 + p82*LCTL4
      + p404*lnapn4
util5 = p1*dist5 + p6*fft5
      + p16*MAPike5 + p17*Tobin5 + p18*Sumner5
      + p171*imldl5 + p172*i2ldl5 + p175*i5ldl5 + p175*i6ldl5
      + p60*d109 + p80*LDL5 + p82*LCTL5
      + p404*lnapn5
util6 = p1*dist6 + p6*fft6
      + p16*MAPike6 + p17*Tobin6 + p18*Sumner6
      + p171*imldl6 + p172*i2ldl6 + p175*i5ldl6 + p175*i6ldl6
      + p60*d124 + p80*LDL6 + p82*LCTL6
      + p404*lnapn6
util7 = p1*dist7 + p6*fft7
      + p16*MAPike7 + p17*Tobin7 + p18*Sumner7
      + p171*imldl7 + p172*i2ldl7 + p175*i5ldl7 + p175*i6ldl7
      + p60*d139 + p80*LDL7 + p82*LCTL7
      + p404*lnapn7
util8 = p1*dist8 + p6*fft8
      + p16*MAPike8 + p17*Tobin8 + p18*Sumner8
      + p171*imldl8 + p172*i2ldl8 + p175*i5ldl8 + p175*i6ldl8
      + p60*d154 + p80*LDL8 + p82*LCTL8
      + p404*lnapn8
util9 = p1*dist9 + p6*fft9
      + p16*MAPike9 + p17*Tobin9 + p18*Sumner9
      + p171*imldl9 + p172*i2ldl9 + p175*i5ldl9 + p175*i6ldl9
      + p60*d169 + p80*LDL9 + p82*LCTL9
      + p404*lnapn9
util10 = p1*dist10 + p6*fft10
      + p16*MAPike10 + p17*Tobin10 + p18*Sumner10
      + p171*imldl10 + p172*i2ldl10 + p175*i5ldl10 + p175*i6ldl10
      + p60*d184 + p80*LDL10 + p82*LCTL10
      + p404*lnapn10
util11 = p1*dist11 + p6*fft11
      + p16*MAPike11 + p17*Tobin11 + p18*Sumner11
      + p171*imldl11 + p172*i2ldl11 + p175*i5ldl11 + p175*i6ldl11
      + p60*d199 + p80*LDL11 + p82*LCTL11
      + p404*lnapn11
util12 = p1*dist12 + p6*fft12
      + p16*MAPike12 + p17*Tobin12 + p18*Sumner12
      + p171*imldl12 + p172*i2ldl12 + p175*i5ldl12 + p175*i6ldl12
      + p60*d214 + p80*LDL12 + p82*LCTL12
      + p404*lnapn12
util13 = p1*dist13 + p6*fft13
      + p16*MAPike13 + p17*Tobin13 + p18*Sumner13
      + p171*imldl13 + p172*i2ldl13 + p175*i5ldl13 + p175*i6ldl13
      + p60*d229 + p80*LDL13 + p82*LCTL13
      + p404*lnapn13
util14 = p1*dist14 + p6*fft14
      + p16*MAPike14 + p17*Tobin14 + p18*Sumner14
      + p171*imldl14 + p172*i2ldl14 + p175*i5ldl14 + p175*i6ldl14
      + p60*d244 + p80*LDL14 + p82*LCTL14
      + p404*lnapn14
util15 = p1*dist15 + p6*fft15
      + p16*MAPike15 + p17*Tobin15 + p18*Sumner15
      + p171*imldl15 + p172*i2ldl15 + p175*i5ldl15 + p175*i6ldl15
      + p60*d259 + p80*LDL15 + p82*LCTL15
      + p404*lnapn15
util16 = p1*dist16 + p6*fft16
      + p16*MAPike16 + p17*Tobin16 + p18*Sumner16
      + p171*imldl16 + p172*i2ldl16 + p175*i5ldl16 + p175*i6ldl16
      + p60*d274 + p80*LDL16 + p82*LCTL16
      + p404*lnapn16
util17 = p1*dist17 + p6*fft17
      + p16*MAPike17 + p17*Tobin17 + p18*Sumner17
      + p171*imldl17 + p172*i2ldl17 + p175*i5ldl17 + p175*i6ldl17
      + p60*d289 + p80*LDL17 + p82*LCTL17
      + p404*lnapn17
util18 = p1*dist18 + p6*fft18
      + p16*MAPike18 + p17*Tobin18 + p18*Sumner18
      + p171*imldl18 + p172*i2ldl18 + p175*i5ldl18 + p175*i6ldl18
      + p60*d304 + p80*LDL18 + p82*LCTL18
      + p404*lnapn18
util19 = p1*dist19 + p6*fft19
      + p16*MAPike19 + p17*Tobin19 + p18*Sumner19
      + p171*imldl19 + p172*i2ldl19 + p175*i5ldl19 + p175*i6ldl19
      + p60*d319 + p80*LDL19 + p82*LCTL19
      + p404*lnapn19
util20 = p1*dist20 + p6*fft20
```

```
      + p16*MAPike20 + p17*Tobin20 + p18*Sumner20
      + p171*imldl20 + p172*i2ldl20 + p175*i5ldl20 + p175*i6ldl20
      + p60*d334 + p80*LDL20 + p82*LCTL20
      + p404*lnapn20
util21 = p1*dist21 + p6*fft21
      + p16*MAPike21 + p17*Tobin21 + p18*Sumner21
      + p171*imldl21 + p172*i2ldl21 + p175*i5ldl21 + p175*i6ldl21
      + p60*d349 + p80*LDL21 + p82*LCTL21
      + p404*lnapn21
util22 = p1*dist22 + p6*fft22
      + p16*MAPike22 + p17*Tobin22 + p18*Sumner22
      + p171*imldl22 + p172*i2ldl22 + p175*i5ldl22 + p175*i6ldl22
      + p60*d364 + p80*LDL22 + p82*LCTL22
      + p404*lnapn22
util23 = p1*dist23 + p6*fft23
      + p16*MAPike23 + p17*Tobin23 + p18*Sumner23
      + p171*imldl23 + p172*i2ldl23 + p175*i5ldl23 + p175*i6ldl23
      + p60*d379 + p80*LDL23 + p82*LCTL23
      + p404*lnapn23
util24 = p1*dist24 + p6*fft24
      + p16*MAPike24 + p17*Tobin24 + p18*Sumner24
      + p171*imldl24 + p172*i2ldl24 + p175*i5ldl24 + p175*i6ldl24
      + p60*d394 + p80*LDL24 + p82*LCTL24
      + p404*lnapn24
util25 = p1*dist25 + p6*fft25
      + p16*MAPike25 + p17*Tobin25 + p18*Sumner25
      + p171*imldl25 + p172*i2ldl25 + p175*i5ldl25 + p175*i6ldl25
      + p60*d409 + p80*LDL25 + p82*LCTL25
      + p404*lnapn25
util26 = p1*dist26 + p6*fft26
      + p16*MAPike26 + p17*Tobin26 + p18*Sumner26
      + p171*imldl26 + p172*i2ldl26 + p175*i5ldl26 + p175*i6ldl26
      + p60*d424 + p80*LDL26 + p82*LCTL26
      + p404*lnapn26
util27 = p1*dist27 + p6*fft27
      + p16*MAPike27 + p17*Tobin27 + p18*Sumner27
      + p171*imldl27 + p172*i2ldl27 + p175*i5ldl27 + p175*i6ldl27
      + p60*d439 + p80*LDL27 + p82*LCTL27
      + p404*lnapn27
util28 = p1*dist28 + p6*fft28
      + p16*MAPike28 + p17*Tobin28 + p18*Sumner28
      + p171*imldl28 + p172*i2ldl28 + p175*i5ldl28 + p175*i6ldl28
      + p60*d454 + p80*LDL28 + p82*LCTL28
      + p404*lnapn28
util29 = p1*dist29 + p6*fft29
      + p16*MAPike29 + p17*Tobin29 + p18*Sumner29
      + p171*imldl29 + p172*i2ldl29 + p175*i5ldl29 + p175*i6ldl29
      + p60*d469 + p80*LDL29 + p82*LCTL29
      + p404*lnapn29
util30 = p1*dist30 + p6*fft30
      + p16*MAPike30 + p17*Tobin30 + p18*Sumner30
      + p171*imldl30 + p172*i2ldl30 + p175*i5ldl30 + p175*i6ldl30
      + p60*d484 + p80*LDL30 + p82*LCTL30
      + p404*lnapn30
util31 = p1*dist31 + p6*fft31
      + p16*MAPike31 + p17*Tobin31 + p18*Sumner31
      + p171*imldl31 + p172*i2ldl31 + p175*i5ldl31 + p175*i6ldl31
      + p60*d499 + p80*LDL31 + p82*LCTL31
      + p404*lnapn31
util32 = p1*dist32 + p6*fft32
      + p16*MAPike32 + p17*Tobin32 + p18*Sumner32
      + p171*imldl32 + p172*i2ldl32 + p175*i5ldl32 + p175*i6ldl32
      + p60*d514 + p80*LDL32 + p82*LCTL32
      + p404*lnapn32
util33 = p1*dist33 + p6*fft33
      + p16*MAPike33 + p17*Tobin33 + p18*Sumner33
      + p171*imldl33 + p172*i2ldl33 + p175*i5ldl33 + p175*i6ldl33
      + p60*d529 + p80*LDL33 + p82*LCTL33
      + p404*lnapn33
util34 = p1*dist34 + p6*fft34
      + p16*MAPike34 + p17*Tobin34 + p18*Sumner34
      + p171*imldl34 + p172*i2ldl34 + p175*i5ldl34 + p175*i6ldl34
      + p60*d544 + p80*LDL34 + p82*LCTL34
      + p404*lnapn34
util35 = p1*dist35 + p6*fft35
      + p16*MAPike35 + p17*Tobin35 + p18*Sumner35
      + p171*imldl35 + p172*i2ldl35 + p175*i5ldl35 + p175*i6ldl35
      + p60*d559 + p80*LDL35 + p82*LCTL35
      + p404*lnapn35
util36 = p1*dist36 + p6*fft36
      + p16*MAPike36 + p17*Tobin36 + p18*Sumner36
      + p171*imldl36 + p172*i2ldl36 + p175*i5ldl36 + p175*i6ldl36
      + p60*d574 + p80*LDL36 + p82*LCTL36
      + p404*lnapn36
util37 = p1*dist37 + p6*fft37
      + p16*MAPike37 + p17*Tobin37 + p18*Sumner37
      + p171*imldl37 + p172*i2ldl37 + p175*i5ldl37 + p175*i6ldl37
      + p60*d589 + p80*LDL37 + p82*LCTL37
      + p404*lnapn37
util38 = p1*dist38 + p6*fft38
      + p16*MAPike38 + p17*Tobin38 + p18*Sumner38
      + p171*imldl38 + p172*i2ldl38 + p175*i5ldl38 + p175*i6ldl38
      + p60*d604 + p80*LDL38 + p82*LCTL38
      + p404*lnapn38
util39 = p1*dist39 + p6*fft39
      + p16*MAPike39 + p17*Tobin39 + p18*Sumner39
      + p171*imldl39 + p172*i2ldl39 + p175*i5ldl39 + p175*i6ldl39
      + p60*d619 + p80*LDL39 + p82*LCTL39
      + p404*lnapn39
util40 = p1*dist40 + p6*fft40
      + p16*MAPike40 + p17*Tobin40 + p18*Sumner40
      + p171*imldl40 + p172*i2ldl40 + p175*i5ldl40 + p175*i6ldl40
      + p60*d634 + p80*LDL40 + p82*LCTL40
      + p404*lnapn40
util41 = p1*dist41 + p6*fft41
      + p16*MAPike41 + p17*Tobin41 + p18*Sumner41
      + p171*imldl41 + p172*i2ldl41 + p175*i5ldl41 + p175*i6ldl41
      + p60*d649 + p80*LDL41 + p82*LCTL41
      + p404*lnapn41
util42 = p1*dist42 + p6*fft42
      + p16*MAPike42 + p17*Tobin42 + p18*Sumner42
      + p171*imldl42 + p172*i2ldl42 + p175*i5ldl42 + p175*i6ldl42
      + p60*d664 + p80*LDL42 + p82*LCTL42
      + p404*lnapn42
```

```
util43 = p1*dist43 + p6*fft43
      + p16*MAPike43 + p17*Tobin43 + p18*Sumner43
      + p171*imldl43 + p172*i2ldl43 + p175*i5ldl43 + p175*i6ldl43
      + p60*d679 + p80*LDL43 + p82*LCTL43
      + p404*lnapn43
util44 = p1*dist44 + p6*fft44
      + p16*MAPike44 + p17*Tobin44 + p18*Sumner44
      + p171*imldl44 + p172*i2ldl44 + p175*i5ldl44 + p175*i6ldl44
      + p60*d694 + p80*LDL44 + p82*LCTL44
      + p404*lnapn44
util45 = p1*dist45 + p6*fft45
      + p16*MAPike45 + p17*Tobin45 + p18*Sumner45
      + p171*imldl45 + p172*i2ldl45 + p175*i5ldl45 + p175*i6ldl45
      + p60*d709 + p80*LDL45 + p82*LCTL45
      + p404*lnapn45
util46 = p1*dist46 + p6*fft46
      + p16*MAPike46 + p17*Tobin46 + p18*Sumner46
      + p171*imldl46 + p172*i2ldl46 + p175*i5ldl46 + p175*i6ldl46
      + p60*d724 + p80*LDL46 + p82*LCTL46
      + p404*lnapn46
util47 = p1*dist47 + p6*fft47
      + p16*MAPike47 + p17*Tobin47 + p18*Sumner47
      + p171*imldl47 + p172*i2ldl47 + p175*i5ldl47 + p175*i6ldl47
      + p60*d739 + p80*LDL47 + p82*LCTL47
      + p404*lnapn47
util48 = p1*dist48 + p6*fft48
      + p16*MAPike48 + p17*Tobin48 + p18*Sumner48
      + p171*imldl48 + p172*i2ldl48 + p175*i5ldl48 + p175*i6ldl48
      + p60*d754 + p80*LDL48 + p82*LCTL48
      + p404*lnapn48
util49 = p1*dist49 + p6*fft49
      + p16*MAPike49 + p17*Tobin49 + p18*Sumner49
      + p171*imldl49 + p172*i2ldl49 + p175*i5ldl49 + p175*i6ldl49
      + p60*d769 + p80*LDL49 + p82*LCTL49
      + p404*lnapn49
util50 = p1*dist50 + p6*fft50
      + p16*MAPike50 + p17*Tobin50 + p18*Sumner50
      + p171*imldl50 + p172*i2ldl50 + p175*i5ldl50 + p175*i6ldl50
      + p60*d784 + p80*LDL50 + p82*LCTL50
      + p404*lnapn50
util51 = p1*dist51 + p6*fft51
      + p16*MAPike51 + p17*Tobin51 + p18*Sumner51
      + p171*imldl51 + p172*i2ldl51 + p175*i5ldl51 + p175*i6ldl51
      + p60*d799 + p80*LDL51 + p82*LCTL51
      + p404*lnapn51

-------------------
- sizes

Size1 = pthsiz1
Size2 = pthsiz2
Size3 = pthsiz3
Size4 = pthsiz4
Size5 = pthsiz5
Size6 = pthsiz6
Size7 = pthsiz7
Size8 = pthsiz8
```

```
Size9 = pthsiz9
Size10 = pthsiz10
Size11 = pthsiz11
Size12 = pthsiz12
Size13 = pthsiz13
Size14 = pthsiz14
Size15 = pthsiz15
Size16 = pthsiz16
Size17 = pthsiz17
Size18 = pthsiz18
Size19 = pthsiz19
Size20 = pthsiz20
Size21 = pthsiz21
Size22 = pthsiz22
Size23 = pthsiz23
Size24 = pthsiz24
Size25 = pthsiz25
Size26 = pthsiz26
Size27 = pthsiz27
Size28 = pthsiz28
Size29 = pthsiz29
Size30 = pthsiz30
Size31 = pthsiz31
Size32 = pthsiz32
Size33 = pthsiz33
Size34 = pthsiz34
Size35 = pthsiz35
Size36 = pthsiz36
Size37 = pthsiz37
Size38 = pthsiz38
Size39 = pthsiz39
Size40 = pthsiz40
Size41 = pthsiz41
Size42 = pthsiz42
Size43 = pthsiz43
Size44 = pthsiz44
Size45 = pthsiz45
Size46 = pthsiz46
Size47 = pthsiz47
Size48 = pthsiz48
Size49 = pthsiz49
Size50 = pthsiz50
Size51 = pthsiz51
```

D.2 Sample GAUSS Program for Generalized CNL

```
/* -------------------------------------------------------------
file://dr-doolittle.mit.edu/d:/users/sramming/cnl/gnleauff.g
----------------------------------------------------------- */

/* Cross-Nested Logit model estimation */
/* from Shlomo 15 May 2001 code */

/* the alphas are not estimated and are given in a separate input file */

  library maxlik,pgraph;
  #include maxlik.ext;
  maxset;

/* global variables */
  NOBS  = 159;  /* number of observations */
  NVAR  =  12;  /* number of independent variables including constants */
  /* and excluding mi and gamma */
  NCOLS = 824;  /* number of columns on input X matrix */
  NALT  =  51;  /* number of alternatives */
  NNEST = 856;  /* number of nests */

/* define input and output files */
/* the input data has to be prepared in a specific sequence */
/* for each coefficient to be estimated, input the observation for each
alternative */
/* the last column in the input data is the choice indicator */
  loadm XMAT = d:\users\sramming\cnl\f4q.fmt;

/* the alpha matrix has to be prepared in a specific sequence */
/* the rows indicate the alternative and the columns indicate the nest */
/* assign a number between 0 and 1 to indicate the proportion of the link to
the route */
/* assign 0 if a link does not belong to a route */
  loadm NMAT=d:\users\sramming\cnl\cnla_fft.fmt;

  output file=d:\users\sramming\cnl\gnleauff.lst reset;
  print "CROSS NESTED LOGIT ESTIMATION - BOSTON DATA SET";
  print "Model 2au: alpha f; PS fu; mu= (1-Sa/Sd) to gamma est";

  /* data manipulation */
  /* ================ */
  YVEC=XMAT[.,52];  /* column 52 of XMAT is the choice index */
  AVAIL=XMAT[.,1:51]; /* detach availability indicators */
  ALTNO=XMAT[.,413]; /* detach number of alternatives for each observation */

/* ------------------------------------
Assemble data matrix in order desired,
dropping extraneous information
------------------------------------ */
  X1=XMAT[.,414:464]; /* distance */
  X2=XMAT[.,359:409]; /* FFT      */
  X3=XMAT[.,465:515]; /* Mass Pike dummy */
  X4=XMAT[.,155:205]; /* Tobin Bridge dummy */
  X5=XMAT[.,516:566]; /* Sumner Tunnel dummy */
  X6=XMAT[.,672:824]; /* ln(delay) for income missing, 2-4, 5-6 */
  X9=XMAT[., 53:103]; /* numbered route time */
  X10=XMAT[.,257:358]; /* least dist and least CT labels */
  X12=XMAT[.,567:617]; /* ln of FFT-based PS */

  XMAT=X1~X2~X3~X4~X5~X6~X9~X10~X12;
  clear X1;
  clear X2;
  clear X3;
  clear X4;
  clear X5;
  clear X6;
  clear X9;
  clear X10;
  clear X12;

  zsum=zeros(NOBS*NNEST,1);
  loadm orig_mi = d:\users\sramming\cnl\cnlm_fft.fmt;

  YPERM = zeros(NOBS,NALT);
  i = 1;
  do while (i <= NOBS);
     YPERM[i,YVEC[i,1]] = 1;  /* indicator of the chosen alternative */
     i = i + 1;
  endo;
  clear YVEC;
  XMAT = XMAT~YPERM;        /* add the indicator of the chosen alternative */
  clear YPERM;

/* initial value for coefficients - use MNL coefficients */
/* ===================================================== */
  B_init = {-0.229,-0.483,-0.369,2.81,1.97,
            -4.36,-0.515,-2.68,0.0799,0.748,0.397,0.630,2};
  /* the nesting coefficient gamma IS estimated */

  B_init = B_init|zeros(NVAR+1,1);
  B_init = B_init[1:(NVAR+1),1];

/* likelihood at zero */
/* ================== */
  B_zero=zeros((NVAR+1),1);
  l_zero=logcnl(B_zero,xmat);

  print "Log-likelihood at zero: " sumc(l_zero);

/* initial likelihood */
/* ================== */
  l_init=logcnl(B_init,xmat);

  print "Initial log-likelihood: " sumc(l_init);

  print " ";

/* log-likelihood function */
proc logcnl(b,xmat);
  local i,m,n,na,prob,v,k,y,ev,denom,vnest,nomin,vn,loglik,
        z1,za,mi1,gam,mi;

  gam = b[(NVAR+1):(NVAR+1),.];
  b = b[1:NVAR,.];

  mi = orig_mi^gam;
  mi = (mi.==zsum)+mi;

  v = zeros(NOBS,NALT);   /* initialize utility function */
  loglik=zeros(NOBS,1);
  y = XMAT[.,(NVAR*NALT+1):(NVAR*NALT+NALT)];  /* detach y */

  mi1=zeros(NNEST,1);
  /* calculate the utility for each alternative */
  k = 1;
  do while k<=NVAR;
  /* multiply each coefficient by the correspondent observ */
     v = v + b[k].*XMAT[.,((k-1)*NALT+1):(k*NALT)];
     k = k+1;
  endo;
  ev=exp(v) .* avail;
  ev=ev';      /* transpose exponent of utilities - dimension [NALT,NOBS] */
  z1=zeros(NNEST,1);
  za=zeros(NALT,1);
  n=1;
  do while n<=NOBS;
    nomin = zeros(NALT,1);
    denom = 0;
    vnest = zeros(NALT,NNEST);
    vn = zeros(NNEST,1);
    m=1;
    do while m<=NNEST;
       vnest[.,m] = ev[.,n] .* NMAT[.,(n-1)*m+m];       /* [NALT,NNEST] */
       mi1[m] = mi[(n-1)*m+m];                          /* detach mi */
       vnest[.,m]=(vnest[.,m]+(vnest[.,m].<=za))^(1/mi1[m])
                    -(vnest[.,m].<=za);
       vn[m]=sumc(vnest[.,m]);                          /* [NNEST,1] */
       nomin = nomin + vnest[.,m]*((vn[m]+(vn[m]<=0))^(mi1[m]-1)
                -(vn[m].<=0));  /* [NALT,1] */
       denom = denom + (vn[m].^mi1[m]);                 /*   [1] */
       m=m+1;
    endo;
    if denom == 0;
       denom = 1;
    endif;
    prob = nomin ./ denom;
    prob = prob+(prob.<=za);
    loglik[n] =  y[n,.] * ln(prob);                   /* [NOBS,1] */
    n=n+1;
  endo;
  retp(loglik);
endp;

/* setting the parameters for the maxlik procedure */
  _max_GradTol = 1e-4;     /* less restrict than Gauss default */
  _max_MaxIters = 20;     /* maximum number of iterations */
  _max_Algorithm = 2;      /* BFGS method, Gauss default */
  __output = 1;           /* display partial results after "output" iterations
*/

  /* invoke maxlik procedure */
  {beta,f,g,cov,retcode} = maxlik(XMAT,0,&logcnl,B_init);
  call maxprt(beta,f,g,cov,retcode);
  print;
  output off;
```

D.3 Sample GAUSS Program for Application of CNL

```
/* -----------------------------------------------------------
file://dr-doolittle.mit.edu/d:/users/sramming/cnl/amys4r_f.g
```

```
------------------------------------------------------------ */

/* Cross-Nested Logit model application */
/* the alphas are not estimated and are given in a separate input file */

/* global variables */
  NOBS  =   5;  /* number of observations */
  NVAR  =  12;
  /* number of independent variables including constants and excluding mi */
  NALT  =  51;  /* number of alternatives */
  NNEST = 900;  /* number of nests */

/* define input and output files */
/* the input data has to be prepared in a specific sequence */
/* for each coefficient to be estimated, */
/* input the observation for each alternative */

/* the alpha matrix has to be prepared in a specific sequence */
/* the rows indicate the alternative and the columns indicate the nest */
/* assign a number between 0 and 1 to indicate the proportion */
/* of the link to the route */
/* assign 0 if a link does not belong to a route */

  load NMAT[NOBS*NNEST,NALT] = d:\users\sramming\cnl\nm_a_fft.txt;

/* data manipulations */

  load JMAT[NOBS,3]=d:\users\sramming\cnl\nm_jn.dat;
  OBSNOS=JMAT[.,1]; /* unique ID for survey respondents */
  ALTNO=JMAT[.,2]; /* detach number of alternatives for each observation */
  /* choice = closest match to observed is in 3rd column */

  ALTLBL={1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23
          24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43
          44 45 46 47 48 49 50 51};
  AVAIL=(ALTLBL .*. ones(NOBS,1)) .<= (ALTNO .*. ones(1,NALT));
  /* create availability indicators */
  clear JMAT;

/* ----------------------------------------
Assemble data matrix in order desired,
dropping extraneous information
------------------------------------ */
  load DMAT[NOBS,52]=d:\users\sramming\cnl\nm_dst.dat;
/*  print DMAT[.,1]'; */ /* check for input errors */
  X1=DMAT[.,2:52]; /* distance */
  clear DMAT;
  load FMAT[NOBS,52]=d:\users\sramming\cnl\nm_fft.dat;
/*  print FMAT[.,1]'; */
  X2=FMAT[.,2:52]; /* FFT      */
  clear FMAT;
  /* the facility dummy matrices are very sparse, and hard coded below */
  X3=zeros(NOBS,NALT); /* Mass Pike dummy */
  X3[5,13]=1;
  X3[5,16:18]=ones(1,3);
  X4=zeros(NOBS,NALT); /* Tobin Bridge dummy */
  /* not surprisingly, all zeros in the blow-up-the-Tobin-Bridge scenario */
  X5=zeros(NOBS,NALT); /* Sumner Tunnel dummy */
  X5[5,11]=1; X5[5,20]=1; X5[5,23]=1; X5[5,26]=1; X5[5,32]=1;

  load CMAT[NOBS,52]=d:\users\sramming\cnl\nm_ct.dat;
/*  print CMAT[.,1]'; */
  CT=CMAT[.,2:52];
  clear CMAT;
  DELAY=CT-X2;
  /* prepare to take log by replacing unavail entries with 1 */
  DELAY=(DELAY .* AVAIL) + ( ones(NOBS,NALT) - AVAIL );
  LNDEL=ln(DELAY);
  clear DELAY;

  load SMAT[NOBS,7]=d:\users\sramming\cnl\ns_incyl.dat;
  /* --------------------------------------------------
  The format of this file is as follows:
  1st column = observation number
  2nd        = income missing dummy (0,1)
  3rd        = income in group 2-4 dummy (0,1)
  4th        = income in group 5-6 dummy (0,1)
  5th        = number of years in Boston
  6th        = learned from maps dummy (0,1)
  7th        = learned by experience dummy (0,1)
  --------------------------------------------------- */

  X6=(SMAT[.,2] .* LNDEL)~(SMAT[.,3] .* LNDEL)~(SMAT[.,4] .* LNDEL);
  /* ln(delay) for income missing, 2-4, 5-6 */

  load RMAT[NOBS,52]=d:\users\sramming\cnl\nm_nrt.dat;
/*  print RMAT[.,1]'; */
  X9=RMAT[.,2:52]; /* numbered route time */
  clear RMAT;

  DST2MIN=(X1 .* AVAIL) + 999*( ones(NOBS,NALT) - AVAIL);
  CT2MIN =(CT .* AVAIL) + 999*( ones(NOBS,NALT) - AVAIL);
  /* prepare for minimization by replacing */
  /* unvailable cells with a large number  */
  MINDST=minc(DST2MIN');
  MINCT =minc(CT2MIN');
  X10=(DST2MIN.==MINDST)~(CT2MIN.==MINCT);
/*  print sumc(X10'); */ /* check for ties */
  /* least dist and least CT labels */
  clear DST2MIN; clear CT2MIN; clear MINDST; clear MINCT;

/*  load PMAT[NOBS,54]=d:\users\sramming\cnl\nm_psf.dat; */
/*  print PMAT[.,1]'; */
/*  PSF=PMAT[.,2:52];
  PSF=(PSF .* AVAIL) + ( ones(NOBS,NALT) - AVAIL);
  X12=ln(PSF); */ /* ln of FFT-based PS */
/*  clear PMAT; clear PSF; */

/* Path Size not needed for this model specification */

  YIB=SMAT[.,5]; /* number years in Boston */
  YMAP=SMAT[.,6];
  YEXPR=SMAT[.,7]; /* learned by experience */
  NK=0.0031*ones(NOBS,1)+0.0031*YIB+0.386*YMAP+0.463*YEXPR;
  clear YIB; clear YMAP; clear YEXPR;
  Y1MNK=(ones(NOBS,1)-NK) .*. ones(1,NALT);
  clear NK;

  YPNRT=X9 ./ CT; /* percent numbered route time */
  Y1MPN=ones(NOBS,NALT) - YPNRT; /* 1 - pct nmrt */

  AIAP=ones(NOBS,NALT) - (Y1MNK .* Y1MPN);
  clear Y1MNK;
  clear CT; clear YPNRT; clear Y1MNP;
  AIAP=(AIAP .* AVAIL) + ( ones(NOBS,NALT) - AVAIL );
  /* replace entries for unavailable alts with one */
  /* gives zero when log taken */
  XIAP=ln(AIAP);
  clear AIAP;

  XMAT=X1~X2~X3~X4~X5~X6~X9~X10~XIAP;
  clear X1;
  clear X2;
  clear X3;
  clear X4;
  clear X5;
  clear X6;
  clear X9;
  clear X10;
/*  clear X12; */

  clear XIAP;

  output file=d:\users\sramming\cnl\amys4r_f.lst reset;
  print "CROSS NESTED LOGIT APPLICATION - BOSTON DATA SET";
  print "Model 4r: alpha f; IAP; no PS - New Mystic Bridge scenario";

  mi = ones(NOBS*NNEST,1);
  i = 1;
  delta = NMAT .>= zeros(NOBS*NNEST,NALT);
  deltasum = sumc((delta'));
  clear delta;
  alphasum = sumc((NMAT'));
  deltasum=deltasum + ( deltasum .== zeros(NOBS*NNEST,1) );
  mi=mi- (alphasum ./ deltasum);
  mi=mi + ( mi .== zeros(NOBS*NNEST,1) );

  clear alphasum;
  clear deltasum;

  NMAT=NMAT'; /* transpose the alpha matrix */

  save d:\users\sramming\cnl\nm_a_fft = NMAT;
  save d:\users\sramming\cnl\nm_m_fft = mi;
/*  loadm NMAT=d:\users\sramming\cnl\cnla_fft.fmt;
  loadm mi  =d:\users\sramming\cnl\cnlm_fft.fmt; */

/* estimated values for coefficients */
  B_est = {-0.242,-0.557,-0.57,2.79,2.09,
           -4.80,-0.183,-2.471,0.0657,1.021,0.875,1.125};
  /* the nesting coefficient is specified by formula */
  probs = pcnl(B_est,XMAT);

  print OBSNOS~probs;

  print;
  output off;

/* log-likelihood function */
proc pcnl(b,xmat);
  local i,m,n, na, prob,v,k,ev,denom,vnest,nomin,vn,
   temp1,temp2,temp3;
  v = zeros(NOBS,NALT);   /* initialize utility function */
  prob = zeros(NOBS,NALT);

  k = 1;
  do while k<=NVAR;
     /* multiply each coefficient by the correspondent observ */
     v = v + b[k].*XMAT[.,((k-1)*NALT+1):(k*NALT)];
     k = k+1;
  endo;
  ev=exp(v) .* avail;
  ev = (ev)';   /* transpose exponent of utilities - dimension [NALT,NOBS] */
  n=1;
  do while n<=NOBS;
    m=1;
    nomin = zeros(NALT,1);
    denom = {0};
    vnest = zeros(NALT,NNEST);
    vn = zeros(NNEST,1);
    do while m<=NNEST;
       na=ALTNO[n];
       i=1;
       do while i<=na;
       vnest[i,m] = ev[i,n] .* NMAT[i,(n-1)*m+m];       /* [NALT,NNEST] */
       if vnest[i,m]>0;
         vnest[i,m] = exp((1/mi[(n-1)*m+m])*ln(vnest[i,m]));
       endif;
       i=i+1;
       endo;
       vn[m] = sumc(vnest[.,m]);                        /* [NNEST,1]  */
       if vn[m]>0;
         nomin =  nomin + vnest[.,m] .* exp((mi[(n-1)*m+m]-1)*ln(vn[m]));
                                           /* [NALT,1] */
         denom =  denom + exp((mi[(n-1)*m+m])*ln(vn[m]));
       endif;
       m=m+1;
    endo;
    prob[n,.] = (nomin ./denom)';          /* [NALT,1] before transposing */

    n=n+1;
  endo;
  retp(prob);
endp;
```

E
Scaling Logit Kernel Parameters

This appendix discusses the issue of scaling utility parameters in Logit Kernel models to be comparable to those of MNL, PSL, C-Logit and CNL models. The results presented in Tables 5-11 and 5-12 scaled the LK coefficients so that the free-flow-time coefficient had the same magnitude as a corresponding MNL or PSL model. This appendix describes an alternative approach based on the relative variances of the Gumbel and Gaussian errors in the models.

Recall the MNL model for route choice:

$$\mathbf{U} = \mathbf{X\beta} + \varepsilon = \mathbf{X\beta} + \mathbf{v}_M ,$$

where ε is a general vector of random error terms, and

ν_M is a vector of i.i.d. Gumbel variables with scale parameter μ_M. (The M subscript refers to MNL.) Therefore, $\text{Var}(\mathbf{v}_M) = (\mathbf{g}/\boldsymbol{\mu}_M{}^2)\mathbf{I}_{J_n}$, where *g* is the variance of a standard Gumbel variable, that is, $\pi^2 / 6$.

Typically μ_M is normalized to one, or alternatively, we can interpret the coefficients we estimate as the product of μ_M and $\boldsymbol{\beta}$.

Now consider the Logit Kernel (LK) model for route choice:

$$\mathbf{U} = \mathbf{X\beta} + \boldsymbol{\varepsilon} = \mathbf{X\beta} + \sigma\mathbf{FT\zeta} + \mathbf{v}_L,$$

where σ is a constant related to the scale of the covariance;

F is the (J_n * M_n) link-path incidence matrix (also written **Δ**);

T is an (M_n * M_n) lower triangular matrix of unknown parameters, which for route choice is taken to be $\mathbf{diag}\left(\begin{bmatrix}\sqrt{l_1} & \sqrt{l_2} & \cdots & \sqrt{l_{M_n}}\end{bmatrix}^{\mathrm{T}}\right)$;

ζ is an (M_n * 1) vector of i.i.d. standard Normal variables; and

ν is a (J_n * 1) vector of i.i.d. Gumbel variables.

Therefore,

$$\mathbf{Cov}(\boldsymbol{\varepsilon}) = \sigma^2\mathbf{FTT}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}} + (\mathbf{g}/\boldsymbol{\mu}_L{}^{\mathbf{2}})\mathbf{I}_{J_n}$$

Now, assuming that **Xβ** offers the same explanatory power in both the MNL and LK models, we would expect the variance of the error term **ε** to be similar in both models. That is,

$$(\mathbf{g}/\boldsymbol{\mu}_M{}^{\mathbf{2}})\mathbf{I}_{J_n} \approx \sigma^2\mathbf{FTT}^{\mathrm{T}}\mathbf{F}^{\mathrm{T}} + (\mathbf{g}/\boldsymbol{\mu}_L{}^{\mathbf{2}})\mathbf{I}_{J_n}.$$

Note that this "equation" is not a precise use of matrix notation. First, The formulation really involves J_n equalities. These equalities are inherently approximate, because the MNL model assumes homoskedasticity (that is, the same μ_{M} for all J_n alternatives), while the LK variance term incorporates heteroskedasticity. Therefore, we may conceptualize this "equation" in much the same way we write an equation representing a regression model. That is, the J_n alternatives each provide some information regarding the relationship between μ_{M} and μ_{L}.

For each alternative i, the above equation is

$$(\mathbf{g}/\boldsymbol{\mu}_M{}^2) \approx \sigma^2 L_i + (\mathbf{g}/\boldsymbol{\mu}_L{}^2) .$$

Let us maintain the convention of normalizing μ_L to one, and estimating μ_M so we can re-normalize the MNL estimates to the LK scale. With some manipulation, this equation becomes

$$\mu_M \approx \sqrt{\frac{\pi^2}{6\sigma^2 L_i + \pi^2}} , \tag{E-1}$$

which we might estimate by taking the mean of this term over all observations and alternatives. Note that $\mu_M \le 1$, and the MNL coefficients can be transformed to the scale of the LK coefficients by dividing μ_M from the MNL estimates (which are really $\mu_M\boldsymbol{\beta}$). Alternatively, the LK coefficients may be transformed to the MNL scale by multiplying by μ_M.

Note that the Path Size coefficient is not scaled, as it represents a multiple (specifically, the number of elementary alternatives) of the scaled utility coefficients. That is,

$$P_n(i) = \frac{\sum\limits_{\text{elem. alts. in } i} \exp(\mu \mathbf{X}_i \boldsymbol{\beta})}{\sum\limits_{j \in C_n} \sum\limits_{\text{elem. alts. in } j} \exp(\mu \mathbf{X}_j \boldsymbol{\beta})} = \frac{PS_i^{\beta_{PS}} \exp(\mu \mathbf{X}_i \boldsymbol{\beta})}{\sum\limits_{j \in C_n} PS_j^{\beta_{PS}} \exp(\mu \mathbf{X}_j \boldsymbol{\beta})} = \frac{\exp(\mu \mathbf{X}_i \boldsymbol{\beta} + \beta_{PS} \ln PS_i)}{\sum\limits_{j \in C_n} \exp(\mu \mathbf{X}_j \boldsymbol{\beta} + \beta_{PS} \ln PS_j)} .$$

Of course, this LK coefficient scaling method is limited by the assumption that the explanatory power of the Gaussian correlations only come from the unexplained part of the corresponding logit model, and not also from $\mathbf{X}\boldsymbol{\beta}$. When this assumption is not true, the variance of the LK model (Gaussian plus Gumbel terms) is larger than the variance of the Gumbel term in the logit-only model. This situation is equivalent to using too large a value of σ in Equation E-1. The result is that μ_M is underestimated, and the transformed LK coefficients will all appear to have a smaller magnitude than the corresponding logit coefficients.

F
Questions to Elicit Geographic Abilities

The following questionnaire was developed in conjunction with Reginald Golledge, John Eliot, Dan Montello and James Marston of the University of California – Santa Barbara; and Galit Toledo and Yoan Anguilet of MIT. This material is based upon work supported by the National Science Foundation under grant No. 9986475 Any opinions, findings, and conclusions or recommendations expressed in this appendix are those of the author(s) and do not necessarily reflect the views of National Science Foundation.

Different versions of the questionnaire were designed for different types of habitual mode use. All respondents received a common set of questions involving classification demographics and general navigation including walking. Auto drivers and passengers received an additional set of questions. Similarly, transit passengers were asked a set of questions relating to the use of buses and subways. Because of the large common segment, we do not present each questionnaire in its entirety. Instead, we present the common questions in section F.1, questions relating to automobile use in section F.2, and transit questions in section F.3. The actual order of these sections can be seen at the web survey site, http://web.mit.edu/its/wayfinding/

F.1 Questions Asked of All Respondents

F.1.1 Classification Questions

1. What is your gender? Female Male

2. Which of the following best describes your level of education?

Some Secondary or High School
High School Diploma or GED
Some College
Associates or Bachelors Degree
Masters or Professional Degree
Doctoral Degree
I'd prefer not to answer

3. Which best describes your educational and career field:

Art or Graphic design
Business or financial services
Clerical
Education
Engineering or architecture
Government or administration
Home maker or childcare provider
Humanities or liberal arts
Manufacturing
Medicine or biomedical sciences
Physical sciences
Retail or services
Social sciences or social services
Technician or skilled crafts
Other (please specify ....)

4. Other than yourself, how many people live in your household?

Of these people, how many are:

Your spouse or partner
Children younger than 5 years old
Children or foster children age range 5 to 15
Children or foster children age 16 or 17
Adult children
Adult not related to you (e.g. roomates, boarders)

5. Which of the following groups includes your age?

under 18
18 to 24
25 to 34
35 to 44
45 to 54
55 to 64
65 and older
I'd prefer not to answer

6. My usual trip is one of the following

Home - Work
Home - School
Home - College or University
Home - Shop
Home - Escort (pick up or drop off another person)

F.1.2 Navigating in a City

This section asks general questions about your preferences and strategies for finding your way around a city. Some questions will ask you to what extent you agree with a statement, and other questions will ask you to rate the difficulty of a particular activity. A few questions will ask you to describe your navigational style by ranking the strategies you are most likely to use.

1. Judging where North is in an unfamiliar city is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

2. I am very good at drawing a map so guests can find my home.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

3. In a group, I usually let others interpret directions.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

4. I always take the same route for my trips.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

5. When I give directions to my house, people tell me they had no problem finding it.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

6. I can point to my home from any place in my home city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

7. Before taking a trip, I always estimate how much time the trip will take.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

8. Producing an accurate sketch map of a familiar neighborhood is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

9. It takes me quite a few visits to a new place to really understand where it is.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

10. When returning to a place I have recently discovered, I enjoy trying to get there from different directions.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

11. I like exploring new areas without maps.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

12. I enjoy reading maps.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

13. Recognizing a familiar building from a different unfamiliar view is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

14. I very easily get lost in a new city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

15. There are many ways to navigate and learn environments. Please rank which techniques you would use to orient yourself in a new environment, where 1 represents your most preferred technique.

Learning where landmarks are located
Learning the major street names
Learning the block numbering system

16. When employed, I am very good at estimating the time it will take to get from my home to my place of work.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

17. It's important for me to know where I am.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

18. I know the location of all the major landmarks in my city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

19. I have very good spatial abilities.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

20. I remember where places are by relating them to my home or work place.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

21. Once I have examined a map, I am very good at remembering where features on the map are located.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

22. It is easy to navigate on road systems that are not regular right-angle grids.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

23. I use buildings rather than natural features to help me remember a route.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

24. On trips home, I have difficulty in recognizing the landmarks or signs I used on the way out.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

25. When planning a trip I make a mental image of the route I have to follow.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

26. When I am an unfamiliar city and get disoriented, I become “stressed out.”

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

27. When I am not pressed for time, I enjoy getting “lost” in a new city and figuring my way back to familiar territory.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

F.1.3 Other Choice Situations and Other Spatial Tasks

This section involves questions about situations other than traveling within a city. Some questions involve navigating between cities, or within a building or shopping mall. Some questions involve choices other than what route to take or what destination to go to. For example, we ask about your shopping habits to investigate whether people give the same type of attention to purchase decisions as they do to travel decisions.

We also ask some questions about spatial skills that don't involve travel, for instance, putting puzzles together. We want to examine whether the skills used in navigating within a city are related to spatial skills for tasks performed at other scales.

Some of these questions may seem odd to you, but please try to answer them as best you can. Your responses will help us establish which types of spatial skills are most related to navigating within a city.

1. I usually let someone else do the navigational planning for long trips.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

2. Estimating the direction of one city in relation to another (e.g., is Miami east or west of Philadelphia?) is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

3. Naming the states which border a given state is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

4. Estimating the distance between cities on a highway map is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

5. When listening to news reports, I often find myself thinking of the location of the country being mentioned.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

6. When landing in a familiar city, I examine road/highway systems to tell from which direction we are landing.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

7. I am very good at putting together a jigsaw puzzle.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

8. I am very good at pointing to an upstairs bathroom I have visited in a building.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

9. I can move around my dwelling in the dark without bumping in to furniture.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

10. I have a poor memory for where I left things.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

11. Walking in the correct direction after getting off an elevator is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

12. I am very good at judging the size of a container to hold leftover food

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

13. I am very good at estimating whether new furniture will fit in a room.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

14. Estimating the distance between two outdoor places when I can see both of them is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

15. Estimating the distance apart of two outdoor places when I can't see them is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

16. I usually visit no more than three different supermarkets each month.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

17. I rarely go to more than three different shops before deciding on a purchase.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

18. I only shop at stores that are close to my home or work.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

19. I always buy my casual clothes at the same shop.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

20. If I discover an ad for a good buy, I don't care how far away the store is.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

21. I prefer to do my indoor recreation as close to my home or workplace as possible.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

22. I prefer one-stop shopping for all my needs.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

23. I have no difficulty in remembering the layout of a shopping center or mall.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

F.1.4 Your Comments

Please tell us any comments, questions or concerns you may have about this survey

Thank you very much for your honest answers to this survey. Your responses will greatly help our research efforts to better understand traveler behavior and way finding strategies.

Please press the following [submit] button only once to submit this survey.

When your responses have been processed, you will see a page saying that an email has been sent, followed by a list of your responses. You may then close your browser, or use its navigation tools to view other locations. Thank you for your participation in this survey!

F.2 Questions Asked of Auto Drivers and Passengers

This section contain questions about your experience traveling in a city by automobile. The first group of questions involves your experiences as a passenger and your familiarity with the roadway networks. If you have a driver's license, you will also be asked about your experiences and opinions as a driver.

1. When walking, I take shortcuts as frequently as possible.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

2. I don't remember routes very well while riding as a passenger in a car.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

3. I have a very good knowledge of the local freeway system

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

4. The only part of the local freeway system I know well lies between my home and workplace.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

5. I know my city's street networks well.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

6. When learning a route through a street network, I try to remember each turn:

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

7. I try to build a mental network of the different routes I learn.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

8. I find it easy to learn different routes.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

9. I find it easy to fit the routes I learn together into a mental map of a network of routes.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

10. Estimating the distance along the road between two familiar locations is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

11. My ability to estimate travel time between two familiar locations is:

Extremely Good 1 2 3 4 5 6 7 8 9 10 Extremely Poor

12. Verbally describing an unfamiliar route that I have just driven is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

13. After driving to a new location using directions given by another person, I can find it again without any help.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

14. Remembering where I parked my car in a large outdoor parking lot is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

15. I find it difficult to remember where my car is parked in a multi-level garage.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

16. I would not feel comfortable navigating a rental car in a new city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

17. I prefer using major surface streets instead of freeways for trips within my home city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

18. Freeway driving is the more dangerous than driving on neighborhood streets:

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

19. Without a map, taking shortcuts in a road network is easy.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

20. When traveling in a road network, I try to take the shortest path to a destination.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

21. When driving, I try to take the fastest path to my destination, even if it is not the most direct or straight-line path.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

22. When driving in a city, I choose routes that maximize the scenic or aesthetic experience of my trip.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

23. When driving in a city, I choose routes which minimize the number of traffic lights on my trip.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

24. When driving in a city, I try to minimize the number of left turns.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

25. When a new section of street or freeway is opened, I immediately drive it to see if I might want to use it in the future will influence my normal route selection practice.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

26. Even if part of a normally traveled driving route is blocked, I find it easy to find another way to my destination.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

27. If my route is blocked by congestion, I prefer to change my destination rather than complete the original trip.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

28. I am very good at judging when it is safe to pass a truck on a narrow, two-lane highway.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

29. Judging when it is safe to turn left at an intersection with traffic coming in the opposite direction is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

30. When driving, judging where the sides of my car are in relation to objects outside the car is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

31. Judging where the curb is when turning right in my car is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

32. Judging when it is safe to merge my car into traffic on a high-speed highway is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

33. After I have driven to a place, I always remember how to get back to it.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

34. I heavily rely on road signs when driving in a city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

35. When driving in a city, I keep updating my position with respect to where the city center is located.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

36. If I drive to several places on the same trip, I only remember where each place is in relation to my last stop.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

37. As I drive in a city, I visualize a map of the city's layout.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

38. When driving, I take shortcuts as frequently as possible.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

39. When taking a trip by car, I prefer to get directions on the way rather than before the start.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

40. Suppose you have a one-hour drive home from work. If traffic on your route was slowed to a crawl, how much time, in minutes, would you wait before taking an alternate route?

5 10 15 20 25 30 wouldn't change

41. Suppose you have a one-hour drive home from work. A 6-month-long road repair project has slowed your trip. How much travel time would you want to save before you would try a new route?

5 10 15 20 25 30 wouldn't change

42. Suppose you have a one-hour trip home from work. Someone at works tells you of a route that is faster than yours. How much travel time would you want to save before trying this entirely new route?

5 10 15 20 25 30 wouldn't change

F.3 Questions Asked of Transit Passengers

F.3.1 Traveling by Public Transportation

This section involves your experiences and attitudes about using public transportation. The first group of questions concerns using public transportation in general. Then there are two sections of questions about using buses and subways, two modes with very different operating characteristics.

1. I find it easy to transfer between different types of public transportation (e.g., from bus to subway).

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

2. When traveling to unfamiliar city, I like to get public transportation information before I leave.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

3. I would consider using public transportation in the city where I live, but not in unfamiliar city.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

4. If I can't find public transportation information when I am planning a trip to an unfamiliar city, I'll plan renting a car instead.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

5. There are many different ways to learn how to make a trip on public transportation. Please rank which methods you would most like to use for planning a trip on public transportation, where 1 represents your most preferred method.

Asking a friend, colleague or family member for written or for verbal instructions.
Looking at the system map and time tables or schedules.
Calling a travel planning phone line.
Visiting a web site that will produce a map of the routes and transfer points I need for my trip.

6. There are also many different ways to learn the fare structure (how much to pay for various types of trips) of a public transportation system in an unfamiliar city. Please rank which methods you would most like to use to learn a new fare structure, where 1 represents your most preferred method.

Asking a friend, colleague or relative who lives in that city.
Asking a bus driver or subway token attendant.
Calling an information line when I travel to that city.
Calling an information line once I arrive in that city.
Seeing the fares summarized in a table on schedules or the system map.

F.3.2 Traveling by Bus

1. While traveling by bus in a city, I like to keep track of where I am.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

2. There are different methods people use to know when to get off a bus in a familiar city. Please rank the methods you most commonly use, where 1 represents the method you use most often.

I tell where to get off by looking out the window for familiar landmarks.
I ask the driver or other passengers to tell me when we come to my stop.
I tell where to get of by judging from how long I have been riding or what turns I have felt the bus make.
I usually pay attention to block numbers.

3. I have no difficulty interpreting bus route maps.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

4. When riding a bus, it's easier to tell where you are if you're traveling on a regular grid network of streets.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

5. It is easy for me to calculate how long my trip will take from bus time tables, even if my stops aren't listed on the table.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

F.3.3 Traveling by Subway

1. While traveling by train in a city, I like to keep track of where I am.

Strongly Agree 1 2 3 4 5 6 7 8 9 10 Strongly Disagree

2. Interpreting a map of a subway system is:

Extremely Difficult 1 2 3 4 5 6 7 8 9 10 Extremely Easy

3. There are different methods people use to know when to get off the subway in a familiar city. Please rank the methods you most commonly use, where 1 represents the method you use most often.

I keep track of how many stops there are until the one I want.
I listen for the name of my station.
I tell where to get of by the decor of the station.
I visualize where on the system map I am.
I visualize where in the city I am.

Colophon

This thesis was prepared with Microsoft Word 97 and the running copy was printed on a Hewlett-Packard LaserJet 4P. Body text is 12 point Times New Roman spaced at one-and-a-half lines within paragraphs, and with 24 points of leading between paragraphs. Equations observe the same line spacing. Up to two variables may be defined after an equation in regular body text. Where three or more variables are defined after an equation, the word “where” is set against the left margin, and variable definitions start at a tab a ½ inch from the left margin, with a ¾-inch hanging indent. Only equations referenced in the text are numbered. Equation numbers have the form of a chapter number or appendix letter, a hyphen, and an increasing equation number. Equation numbers are set off in parentheses flush right from the equation they reference.

Chapter, appendix and other major titles are right-justified 36 point Univers bold followed by 24 points of leading. Chapter numbers and appendix letters are right-justified 36 point Univers. Section headings are 14 point Univers bold, and section subheadings (e.g., 2.1.3) are 14 point Univers. Both section headings and subheadings are preceded by 12 points of leading and followed by 10 points of leading. Page headers and footers are 12 point Univers, justified so that text is flush with the outside of the page. Headers are separated by a 1-point line, while footers are separated by a 2-point line.

Chapter, appendix and other major entries in the table of contents are 12 point Univers Bold. Section heading, section subheading, table and figure entries are 12 point Univers.

Text of the references is set in 11 point Times New Roman, leaded to 14 point, with an additional 6 points between entries. Entries have a ¼-inch hanging indent.

Captions of tables and figures are 14 point Univers bold, centered. Subcaptions for parts of tables or figures are set in 12 point Univers. Table headings are Univers bold, against a 20 percent grey background. Table body text is set in Univers. Major table elements are demarcated with 2¼-point lines, and table cells are separated by 1-point lines. Notes and sources are 10 point Univers, with 10-point leading between the table or figure, notes, sources and the caption. Notes and sources begin at ¾ inches from the left margin and have a 1-inch hanging indent.